Multi-Label Text Classification by Graph Neural Network with Mixing Operations

Anonymous ACL submission

Abstract

Multi-label text classification is one of the fundamental tasks in natural language processing. Recently, the graph convolution network (GCN) is leveraged to boost the performance of such a task. However, the best way for label correlation modeling and feature learning with label system awareness is still unclear. This paper proposes Mix-GCN, a graph network with two mixing operations, to improve the conventional GCN framework for multi-label text classification in the following two steps. Firstly, we model the label correlations by mixing the graph built from statistical co-occurrence information and the graph constructed from prior knowledge. Secondly, we propose a mixing operation to continuously inject GCN embedding into LSTM representation learning for better label-aware representation. Experimental results on four benchmarks demonstrate that Mix-GCN significantly outperforms the state-of-the-art models and performs better in long-tail label cases.

1 Introduction

Multi-label is a universal property of data; it is common for a text, image, or video to have a multi-label whose contents are connected to multiple domains. Multi-label text classification (MLTC) is a fundamental and practical issue in natural language processing, which is gaining more academic interest as the amount of data rises. MLTC has been the subject of numerous recent research, with significant advancements in web mining (Agrawal et al., 2013; Jain et al., 2016), sentiment analysis (Huang et al., 2013; Yu et al., 2018), and information retrieval (Zhao et al., 2015; Ranjan et al., 2015). It is, nevertheless, an unsolved and challenging process due to the vast categories of classification and their intricate interactions between labels.

There are two paradigms for MLTC research: learning enhanced document representation (Liu et al., 2017; Yang et al., 2018) and modeling label correlation (Chen et al., 2017; Zhang et al., 2018; Adhikari et al., 2019). Both of them looked at informative terms in text content, label structure, and semantics to capture label correlations and category information to discover label-specific components. Despite their success, they face two challenges.

The first problem lies in label correlation modeling, a common way to explore the label correlations in the document is to utilize the statistical correlations between categories to build a label co-occurrence graph for guiding interactions. Most of the previous methods (Du et al., 2019; Xiao et al., 2019; You et al., 2019) learn the same document representations for different labels; they do not explicitly consider the corresponding semantic parts of each label in the document. Recent study (Ma et al., 2021) has employed attention mechanisms and GCN to investigate the above semantic connections and learn a label-specific text representation for MLTC. In MLTC, such a strategy outperformed non-GCN methods, demonstrating the importance of using a graph neural network to model label correlation.

However, such a graph is insufficient because it is based solely on statistical co-occurrence information. There are two main reasons for this: Firstly, the co-occurrence information collected from the training set is insufficient. Label co-occurrences in the test set but not in the training set, for example, may be missed. Furthermore, statistical co-occurrence may result in the formation of a long-tail distribution. As a result of this phenomenon, models may be unable to predict long-tail labels.

The second problem exists in the document representation learning. Formally, given text X, the traditional method in MLTC for predicting its labels can be generally formulated as a two-stage procedure: (1) feature extraction process that encodes the document. (2) a label statistical co-occurrence graph to guide the representation learning. Therefore, the GCN embeddings are only explicitly in-

volved once as supervision in the training phase. All the works mentioned above follow the conventional practice of two-stage procedure, and the full structure of the label system is neglected in representation learning.

To address the issues above, in this paper, we propose Mix-GCN for MLTC. Specifically, 'Mix' refers to two designs in our model:

- Instead of only using a statistics co-occurrence graph to build label relations, we add an extra knowledge-based graph to mix both graphs into the final one by a convex combination. Such a knowledge-based graph can assist the model to realize the real-world relations of the labels; even such relations do not occur in the training set.

- To learn better feature representations for an MLTC task anchored on its label structures, we design a mixing procedure for LSTM and GCN networks to inject GCN embeddings to LSTM for the label-aware representation learning procedure. Specifically, We constantly create mixing operations between LSTM and GCN at every layer to inject label information into LSTM, unlike previous approaches that only include label information once.

The contributions of this paper are summarized as follows:

- We propose a graph mixing framework (Mix-GCN) for multi-label text classification that takes advantage of the entire label system's embedding for representation learning by establishing layers between GCN and LSTM for label-aware representation learning.

- We construct the graph in Mix-GCN based on statistical co-occurrence information and label knowledge priors to model the correlations between labels accurately and comprehensively.

- We launch experiments on four benchmarks, and the results show that our model Mix-GCN outperforms state-of-the-art models and achieves better performance with respect to tail labels. Besides, ablation studies validate the effectiveness of two mixing operations in our Mix-GCN.

2 Related Work

2.1 Multi-label Text Classification

The extant MLTC approaches mainly concentrate on learning more comprehensive document representations (Liu et al., 2017) and label-correlation modeling (Nam et al., 2017; Yang et al., 2018; You et al., 2019) to enhance performance.

With the extensive employment of neural network methods in text representation, various innovative models have been proposed, including traditional deep learning and Seq2Seq-based methods. Liu et al (Liu et al., 2017) used CNN and dynamic pooling to learn text representation. However, they treat all words equally, thus failed to capture informative words in documents. As for seq2Seq methods, such as MLC2Seq (Nam et al., 2017) and SGM (Yang et al., 2018), they used RNNs to encode the input text and an attention-based RNN decoder to generate predictive labels sequentially. Although they use an attention mechanism to capture information words in the text content, these models cannot distinguish similar labels well since they ignored the semantic connection between labels and documents and learn the same document representations for different tags.

Recently, some studies (You et al., 2019; Xiao et al., 2019; Chalkidis et al., 2019) used attention mechanisms to explore the interactions between words and labels and learned a document representation for a specific label. These methods have achieved promising performance, which confirms the importance of exploring semantic connections.

Our work mostly relates to the proposed LDGN (Ma et al., 2021), which used a dual-GCN to propagate information among labels and merges label information with document representation in the final stage. Differently, our Mix-GCN builds the adjacency matrix of GCN by mixing the statistical co-occurrence graph and the knowledge-based graph. Moreover, the label information from GCN is continuously absorbed into the document representation in LSTM for better feature learning.

3 Methodology

3.1 Overview

In this paper, we propose a mixing framework for multi-label text classification called Mix-GCN. We provide a novel label correlation modeling technique by combining the statistical label graph and the prior-oriented label graph. Then a better fea-

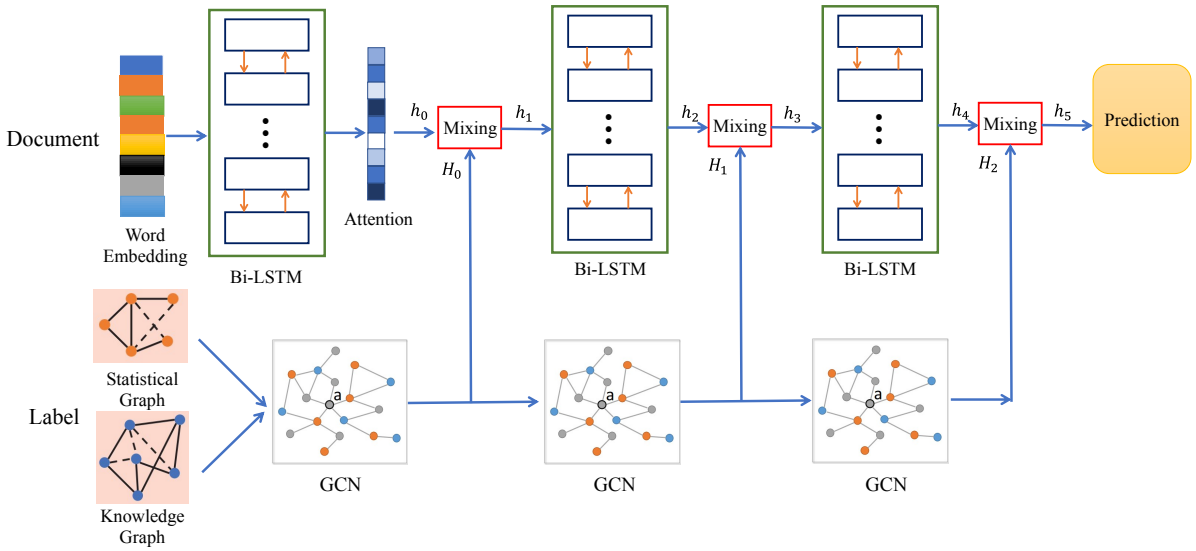


Figure 1: Overview of Mix-GCN.

ture learning operation is developed by absorbing label structure information provided by GCN at the layers of LSTM, and Figure 1 shows the overview of Mix-GCN.

3.2 Problem Formulation

Denote $D = \{(x_i, y_i)\}_{i=1}^{N}$ as the training set, which consists of N samples with corresponding labels $Y = \{y_i \in \{0,1\}^l\}$; here, l is the total number of label categories. Every sample document is composed of several words. Each word can be represented as a k -dimension vector through Glove (Pennington et al., 2014). Denote $x_i = \{w_1, \cdots, w_j, \cdots, w_n\}$ as the contents in the i -th document, $w_j \in \mathbb{R}^k$ represents the embedding of j -th word in the document, k is the embedding size, and n indicates the number of words. Specifically, in text classification, a label is also a word that contains textual information. Therefore, one label can also be encoded as an embedding vector, and the label set will be represented by a matrix $C \in \mathbb{R}^{l \times k}$. Multi-label text classification targets to learn the mapping from the input text sequence to the most relevant labels.

3.3 Document Representation

We first embed each word in the text into a word vector $w_j \in \mathcal{R}^k$, where k is the dimension of word embedding, given a document x with n words. We employ a bi-LSTM to capture word-level semantics for improved document representation to collect contextual information from both directions of the text sequence. To acquire the final document representation $\mathbf{h}$, we concatenate the backward and forward hidden states.

Then, to acquire the document representation for a certain label, we utilize a basic attention technique to obtain the relevant semantic components associated with each label. First, we use the word-corresponding vectors in Glove to establish the label representation $C \in R^{l \times k}$, and then we compute the attention values anchored on the document representation $\mathbf{h}$. The semantic components of a given label can be generated depending on the attention directed by the label, which can be formally described as follows:

$$att_{ij} = \frac{e^{(\mathbf{h_j c_i})}}{\sum_j e^{(\mathbf{h_j c_i})}} \quad (1)$$

$$u_i = \sum_j att_{ij} \mathbf{h_j} \quad (2)$$

Among the variables, att_{ij} indicates how informative the j -th text feature vector is for the i-th label in the label set C, and $u_\mathbf{i} \in R^D$ represents the label-specific representation anchored on label $\mathbf{c_i}$ for this document.

3.4 Mixing in Graph Construction

In Mix-GCN, our graph is constructed by mixing the statistical labeled graph and the knowledge-oriented graph through convex combinations. The statistical graph in this paper is defined as the graph constructed with statistical information such as the label co-occurrence of such labels. The statistical information is determined by the distribution of

samples in the training set, so it is highly dependent on the completeness of the training set. As mentioned in the introduction, statistical graphs are significantly affected by noise and neglect in the training set. Meanwhile, knowledge graphs, such as ConceptNet (Speer et al., 2017), are established with human knowledge in several ways, such as resources created by experts, thus may complement the statistical graph.

The knowledge graph contains real-world knowledge for representing the relationship of labels, thus may help model learn the relation between labels, especially when the relations do not appear in the training set. However, it has three drawbacks: (2)The graph is very dense and it contains too many unnecessary node relationships. When used for deeper GCNs, it has a more negative impact on over-smoothed label embeddings than sparse graphs. (2) It is independent of the dataset and the task and therefore neglects task-specific or dataset-specific features. (3) It does not include all the labels in the dataset, so the edges of such labels are missed, leading to poor performance on these labels.

Our Mix-GCN combines both statistical information and prior knowledge to overcome their disadvantages and utilize their advantages. We will formally illustrate the details as follows.

Let $G = (V, E, A)$ as a standard graph, where V, E, A denote nodes, edges and adjacency matrix of G. A is an $N \times N$ matrix with (i, j) entry representing the weight of edges between nodes V_i and V_j, where $N = |V|$ is the number of vertices. $E \in R^{N \times k}$ denotes the label embedding matrix for all N nodes.

Then, let $G_C = (V, E_C, A_C)$ denote the statistical graph, and $G_P = (V, E_P, A_P)$ as knowledge graph, where A_C and A_P are adjacency matrices obtained from statistical information and human knowledge, respectively. A_C is obtained by the method in (Chen et al. 2019), and A_P is constructed by the expert-created ConceptNet (Speer, Chin, and Havasi 2017). Specifically, the nodes V in G_P represent the labels (e.g. science) in label set C. The construction of A_P can be defined as follows:

$$A_{Pij} = \begin{cases} \max\{score_r \mid r \in R_{ij}\}, & \text{if } |R_{ij}| > 0 \\ 0, & \text{if } |R_{ij}| = 0 \end{cases} \tag{3}$$

where R_{ij} is a set of relations (e.g., 'similar') between nodes extracted from ConceptNet. $score_r$ is the weight of relation r. $|R_{ij}|$ is the number of elements in R_{ij}.

Denoting A'_C and A'_P as the normalized versions of A_C and A_P, respectively. $A'_C = D_C^{-1/2} A_C D_C^{-1/2}$, where D_C is diagonal and $[D_C]_{ii} = \sum_j [A_C]_{ij}$, and A_P is normalized similarly. Then a convex combination of A'_C and A'_P is used to mix the knowledge graph and statistical graph and the new adjacency matrix A^* is defined as follows:

$$A^* = \lambda A'_C + (1 - \lambda) A'_P \tag{4}$$

where $\lambda \in [0, 1]$ is a weight hyper-parameter. Meanwhile, because the elements of A'_C and A'_P are non-negative, A^* has more non-negative elements compared with A_C and A_P. In other words, the graph constructed with A^* has more unnecessary edges than G_S or G_K, as shown in Figure 3. To decrease such edges, we use a threshold α to filter the elements.

$$[A_\alpha]_{ij} = \begin{cases} 0, & \text{if } A^*_{ij} < \alpha \\ A^*_{ij}, & \text{if } A^*_{ij} \geq \alpha \end{cases} \tag{5}$$

As claimed in previous works, when the number of GCN layers increases, the performance of models decreases in some tasks. The phenomenon is possibly due to the over-smoothing of deeper GCN layers (Chen et al., 2019). Spurred by such findings, we further modify the entries in the adjacency matrix of the mixed graph and get the final adjacency matrix A_F :

$$A_F = \beta A_\alpha + (1 - \beta) I$$

where I is an identity matrix. β is also a hyper-parameter that determines the weights. Based on the adjacency matrix A_F, we construct the edges as:

$$E_F = \left\{ (V_i, V_j) \mid [A_F]_{ij} \neq 0, \text{ and } 0 \leq i, j \leq N \right\} \tag{6}$$

(V_i, V_j) denotes the edge of nodes V_i and V_j. The graph we proposed is defined as $G_F = (V, E_F, A_F)$, which is called final graph.

3.5 Mixing between GCN and LSTM

Then, based on the final graph G_F, we use GCN (Kipf and Welling, 2016) to understand the deep connections between label-specific semantic components. GCNs are graph-based neural networks that can improve node representations by propagating messages between nearby nodes.

Dataset	N	M	D	L	L_1	L_2	W	W^*
RCV1	23,149	781,265	47,236	103	3.18	729.67	259.47	269.23
AAPD	54,840	1,000	69,399	54	2.41	2444.04	163.42	171.65
EUR-Lex	11,585	3,865	171,120	3,956	5.32	15.59	1,225.20	1,248.07
Kanshan-Cup	2,799,967	200,000	411,721	1,999	2.34	3513.13	38.06	35.48

Table 1: N is the amount of training samples; M is the amount of test samples; D is the total amount of words, L is the total amount of classes; L_1 is the average amount of labels per sample; L_2 is the average amount of samples per label; W is the average amount of words per sample in the training set; W^* is the average amount of words per sample in the testing set.

In the GCN, the label embeddings of each node is a weighted sum of the embeddings of its neighbors from the previous layer. We follow a common practice as was done in (Gao et al., 2018; Wu et al., 2019) to apply graph convolution:

$$H^{(l+1)} = \sigma\left(A'_F H^{(l)} W^{(l)}\right)$$

where A'_F is the normalized adjacency matrix. $H^{(l)}$ denotes the label embedding at the l -th layer in a GCN. Note that $H^{(0)}$ is the initial word embeddings of labels. $W^{(l)}$ is a learnable matrix in the training phase. $\sigma(\cdot)$ denotes the LeakyRelu activation function.

Rather than providing label relationship information to representation all at once, we suggest injecting label information into LSTM at multiple phases via mixing procedures. In our Mix-GCN, a mixing operation is defined as follows:

$$h_{l+1} = (\sigma(H^l) \otimes h_l) \cdot W + h_l \tag{7}$$

where h_{l+1} is output of the mixing mechanism which will be fed to next LSTM, H^l is the hidden label embeddings of GCN, h_l is the document representation of the current LSTM, and W is the learnable matrix that ensures the mixing mechanism keeps the shape of h_{l+1} the same as h_l. Specifically, h_0 is the initial document embedding U defined in Eq.(2).

The mixing procedure is designed to encourage the LSTM to learn label-system anchored feature representations to improve representation learning. It calculates the dot product between features and label embeddings, which shows how each feature point is related to a label embedding. The mixing procedure links the label system and the LSTM representation, and the learned representation is label-aware.

The mixing procedure has two principle advantages. (1) GCN embeddings can help LSTM feature learning by making the LSTM representation aware of label relationships. (2) The extra gradients from the mixing operation may be regarded as a particular regularization in the hidden embeddings learning process, forcing hidden embeddings to adapt to representation more properly. To a certain extent, it can deal with the over-smoothing problem of deep GCNs.

After the above procedures, we concatenate the two representation $\mathbf{H}^* = \left[\mathbf{h}^3, \mathbf{h}^5\right]$ and feed it into a FFN for prediction with the multi-label cross entropy loss:

$$\mathcal{L} = \sum_{c=1}^{C} y^c \log\left(\widehat{y}^c\right) + \left(1 - y^c\right) \log\left(1 - \widehat{y}^c\right)$$

where y^c and y represent the prediction and ground-truth label, respectively.

4 Experiment

4.1 Benchmarks

In this paper, four benchmarks are used to construct the experiments.

- **RCV1**: it contains more than 80K manually categorized news belonging to 103 classes (Lewis et al., 2004).
- **AAPD**: it collects the abstract and the corresponding subjects of 55840 papers from arXiv in the filed of computer science (Yang et al., 2018).
- **EUR-Lex**: it is a collection of documents about European Union law belonging to 3956 subjects. The public version3 contains 11585 training instances and 3865 testing instances (Mencia and Fürnkranz, 2008).
- **KanShan-Cup**[1]: it is released by the largest Chinese community question answering plat-

[1]https://www.biendata.xyz/competition/zhihu/data/

form, Zhihu. It contains near 3 million questions about 1999 topics.

4.2 Evaluation Metrics

Following the settings of previous work (You et al., 2019; Xiao et al., 2019), we use precision at top K (P@k) and Normalized Discounted Cumulated Gains at top K (N@k) for performance evaluation. The definition of two metrics can be referred to You et al. (2019)

4.3 Baselines

- **XML-CNN** (Liu et al., 2017): it adopts CNN and a dynamic pooling technique to extract high-level feature for multi-label text classification.
- **SGM** (Yang et al., 2018): it applies a sequence generation model from input document to output label to construct the multi-label text classifier
- **DXML** (Zhang et al., 2018): it tries to explore the label correlation by considering the label structure from the label co-occurrence graph.
- **AttentionXML** (You et al., 2019): it builds the label-aware document representation only based on the document contents with a probabilistic label tree and multi-label attention.
- **EXAM** (Du et al., 2019): a novel framework that leverages the label information to compute the word-level interactions.
- **LSAN** (Xiao et al., 2019): a label-specific attention network model based on self-attention and label-attention mechanism.
- **LDGN** (Ma et al., 2021): it adopts a Dual-GCN to incorporate category information to learn label-specific components from documents.

4.4 Implementation

We adopt 300-d GloVe (Pennington et al., 2014) to generate the initial embeddings of words and labels. As for the labels whose names are out-of-vocabulary (OOV) in GloVe, we use the average embeddings of all labels as the representation. We set λ in (4) to be 0.1, α in (5) to be 0.03 and β in (6) to be 0.3. Adam is used as the optimizer with a momentum of 0.9, weight decay of 104 and batch size of 16. The initial learning rate of Adam is 0.001 and the model trained for 80 epochs in total.

4.5 General Results

Tables 2 and 3 show the results of all of the comparative approaches in the four benchmarks. The experimental results of baseline models are explicitly quoted from prior works for a fair comparison.

Tables 2 and 3 show the results on four different datasets; the proposed Mix-GCN outperforms all other baselines. The excellent results validate the effectiveness of mixing procedure learning with dual graph neural networks, including two components: (1) graph construction based on statistical graph and knowledge graph, and (2) representation mixing between GCN and LSTM. The performance of XML-CNN is found to be inferior to that of other methods of comparison. This is because it only uses the text content of documents to classify them, ignoring the label correlations, which are crucial in multi-label classification. AttentionXML, a label tree-based model, outperforms the seq2seq method (SGM) and the deep embedding method (DXML). Although DXML and SGM use a label graph or an ordered sequence to model label relationships, they ignore interactions between labels and document content. LSAN also employs multi-label attention, which focuses on the most important parts of the content while extracting different semantic information for each label.

Specifically, LDGN outperforms other label attention-based methods because it uses a dual graph network with adaptive fusion to integrate attention and label co-occurrence to learn the label-specific document representation, which takes into account the semantic correlations between document content and labels text.

Generally, our Mix-GCN outperforms sequence-to-sequence, deep embedding, and label attention-based models, and the MLTC metrics P@k and nDCG@k improve significantly. On AAPD dataset, Mix-GCN improves P@1 of LDGN method from 86.24% to 86.98% and enhances nDCG@3 and nDCG@5 from 83.33% to 84.02%, 86.85% to 87.43%, respectively. As for EUR-Lex dataset, the metric P@1 is increased from 81.03% to 82.11%, and nDCG@3 and nDCG@5 are improved from 71.81% to 72.68%, 66.09% to 68.01%, respectively. On RCV1 dataset, P@1 increases by 0.8%, and Mix-GCN achieves 0.62% and 1.1% improvements on nDCG@3, 5 compared with LDGN. The proposed Mix-GCN model's improvements show that both carefully designed mixing mechanisms are generally helpful and effective, and Mix-GCN can

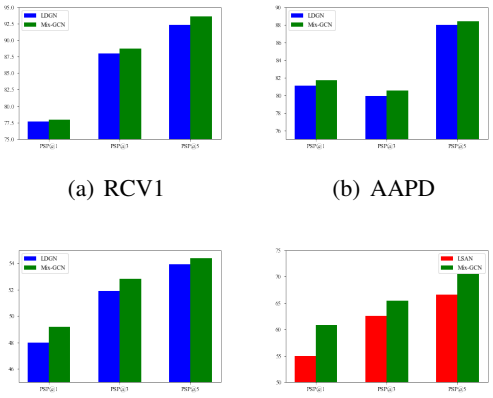
Model	AAPD					EUR-Lex				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
XML-CNN	74.38	53.84	37.79	71.12	75.93	70.40	54.98	44.86	58.62	53.10
SGM	75.67	56.75	35.65	72.36	75.35	70.45	60.37	43.88	60.72	55.24
DXML	80.54	56.30	39.16	77.23	80.99	75.63	60.13	48.65	63.96	53.60
AttentionXML	83.02	58.72	40.56	78.01	82.31	67.34	52.52	47.72	56.21	50.78
EXAM	83.26	59.77	40.66	79.10	82.79	74.40	61.93	50.98	65.12	59.43
LSAN	85.28	61.12	41.84	80.84	84.78	79.17	64.99	53.67	68.32	62.47
LDGN	86.24	61.95	42.29	83.32	86.85	81.03	67.79	56.36	71.81	66.09
Mix-GCN	**86.98**	**62.56**	**42.97**	**84.02**	**87.43**	**82.11**	**69.02**	**57.22**	**72.68**	**68.01**

Table 2: Comparisons between state-of-the-art methods on AAPD and EUR-Lex datasets. The bold numbers indicate the best performance.

Model	RCV1					Kanshan-Cup				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
XML-CNN	95.75	78.63	54.94	89.89	90.77	49.68	32.27	24.17	46.65	49.60
SGM	94.04	78.65	54.38	89.83	90.21	50.84	32.69	24.07	49.54	52.16
DXML	95.37	81.36	53.06	91.76	90.69	50.32	31.83	23.95	46.90	50.47
AttentionXML	96.41	80.91	56.38	91.88	92.70	53.69	34.10	25.16	51.03	53.96
EXAM	93.67	75.80	52.73	86.85	87.71	51.41	32.81	24.29	49.32	49.74
LSAN	96.81	81.89	56.92	92.83	93.43	54.46	34.56	25.54	51.43	54.36
LDGN	97.12	82.26	57.29	93.80	95.03	-	-	-	-	-
Mix-GCN	**97.98**	**82.56**	**58.97**	**94.42**	**96.13**	**57.61**	**36.02**	**27.02**	**52.68**	**55.01**

Table 3: Comparisons between state-of-the-art methods on RCV1 and Kanshan-Cup datasets. The bold numbers indicate the best performance.

capture more comprehensive correlations between categories than LDGN.



(a) RCV1 (b) AAPD

(c) EUR-Lex (d) Kanshan

Figure 2: Comparsion on tail labels.

4.6 Results on tail labels

To investigate the performance of Mix-GCN in tail-label cases, we[2] evaluate Mix-GCN by propensity scored precision at k (PSP@k), which is defined as follows:

$$PSP@k = \frac{1}{k}\sum_{l=1}^{k}\frac{y_{\text{rank}(l)}}{P_{\text{rank}(l)}}$$

Details of PSP@K can be found in (Jain et al., 2016; Ma et al., 2021). As shown in Figure 2, the proposed Mix-GCN performs better in predicting tail labels than the LDGN model (the best baseline) on three datasets. Specifically, on the RCV1 dataset, LDGN achieves 0.96% and 1.40% absolute improvement in terms of P SP@3 and P SP@5 compared with LDGN. On the AAPD dataset, the P SP@k increased by at least 0.53% up to 0.70%. Moreover, on the EUR-Lex dataset, LDGN achieves 1.74%, 3.55%, and 3.03% absolute improvement on P SP@1, 3, 5 compared with LDGN. The improvement in the EUR-Lex dataset is more obvious because label-aware representation learning is more useful for capturing related information in a benchmark with numerous labels. The results prove that Mix-GCN can effectively alleviate the problem of predicting tail labels.

5 Ablation

5.1 Influence of Hyper-parameters

This experiment is conducted on AAPD. When λ varies from 0 to 0.5 by step of 0.1 and

[2]We compare Mix-GCN to LSAN on Kanshan-Cup benchmark instead of LDGN because LDGN is not evaluated on Kanshan-Cup.

keep other parameters as described above, P@1 is $86.65, 86.98, 86.44, 86.23, 86.03$ and 85.59. When we fix λ as 0.1, α varies from 0.01 to 0.04 by step of 0.01, P@1 is $85.60, 86.22, 86.98$ and 86.33. When β varies from 0 to 0.5, P@1 is $85.12, 85.82, 86.44, 86.98, 85.84$ and 85.29.

5.2 Influence of Graph Construction

In order to evaluate the influence of mixing two graphs, we implement three versions of Mix-GCN with statistical graph, knowledge graph, and our proposed mixing graph. They are all built on the same framework, which consists of three GCN layers. The results of applying the statistical graph G_C, knowledge graph G_P, and mixing graph G_F are summarized in Table 4 [3]. Experiments demonstrate that knowledge graph G_P performs worse than statistical graph G_C and mixing graph G_F, which is due to the missing relationships of uncovered labels in the knowledge graph and the over-smoothing impact introduced by many trivial edges. In AAPD, knowledge graph G_P performs much worse than the other two paradigms. The labels in AAPD are more specific (e.g., cs.ce) thus most of them do not appear in the knowledge graph. Therefore, the graph constructed from knowledge is not reliable, and G_P results in poor performance. Furthermore, the statistical graph G_C performs worse than the mixing graph G_F because of the lack of prior knowledge on the four benchmarks. Overall, experiments show that our mixing graph outperforms the two methods, validating the effectiveness of mixing statistical and knowledge graphs.

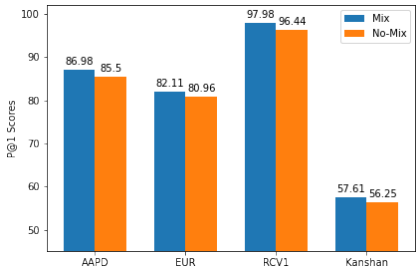


Figure 3: Comparison between 'Mix' and 'No-Mix'

5.3 Influence of Layers in Mix-GCN

In this experiment, we modify the number of layers in Mix-GCN. Specifically, the layer here refers to LSTM+GCN+Mixing (e.g., three layers are in the architecture shown in Figure 1). Experimental results are shown in Table 5[4]. Mix-GCN (3 layers) achieves better performance than Mix-GCN (2 layers) and Mix-GCN (4 layers) by P@1, P@3, and P@5 average improvements over 1.3%, 1.15%, and 1.35% in all benchmarks. Similarly, Mix-GCN (3 layers) obtains the best performance on N@3 and N@5. Specifically, when GCN has no less than two layers, as reported in ML-GCN (Chen et al., 2019), the performance of conventional GCN degrades as the number of GCN layers increases. To some extent, our model alleviates this problem. This is because (1) more GCN layers mean more Mixing operations, which help LSTM learn better label-aware features. (2) the mixing operation contains a skip connection, which can be regarded as a regularization when GCN learns representation.

5.4 Influence of Mixing between LSTM and GCN

In this experiment, we evaluate the effectiveness of continuous mixing operation between GCN and LSTM. Specifically, we only add the mixing in the final layer (third layer) and denote the setting as 'No-Mix.' As shown in Figure 3, 'Mix' performs better than 'No-Mix' on all benchmarks. The results demonstrate the effectiveness of establishing a mixing operation between GCN and LSTM at each layer.

6 Conclusion

In this paper, we propose Mix-GCN, which consists of two mixing operations. Firstly, it mixes the knowledge graph and the statistical graph for label relation modeling. Then another mixing operation is designed for injecting GCN embeddings into LSTM representation, resulting in a label-aware representation learning for Mix-GCN, which acts as label-feature correlation modeling and helps the model learn label-anchored feature representations. Our Mix-GCN is shown to be capable of learning better feature representations for a specific multi-label text classification anchored on its label relationship. Experiments on four benchmarks validate that Mix-GCN achieves state-of-the-art performance in multi-label text classification.

References

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. 2019. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*.

[3]The table is deferred to the appendix due to limited space

[4]The table is deferred to the appendix due to limited space

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24.

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Large-scale multi-label text classification on eu legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322.

Guibin Chen, Deheng Ye, Zhenchang Xing, Jieshan Chen, and Erik Cambria. 2017. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In *2017 international joint conference on neural networks (IJCNN)*, pages 2377–2383. IEEE.

Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. 2019. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5177–5186.

Cunxiao Du, Zhaozheng Chen, Fuli Feng, Lei Zhu, Tian Gan, and Liqiang Nie. 2019. Explicit interaction model towards text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6359–6366.

Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. 2018. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1416–1424.

Shu Huang, Wei Peng, Jingxuan Li, and Dongwon Lee. 2013. Sentiment and topic analysis on social media: a multi-task multi-label classification approach. In *Proceedings of the 5th annual ACM web science conference*, pages 172–181.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 115–124.

Qianwen Ma, Chunyuan Yuan, Wei Zhou, and Songlin Hu. 2021. Label-specific dual graph neural network for multi-label text classification. pages 5419–5429.

Eneldo Loza Mencia and Johannes Fürnkranz. 2008. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 50–65. Springer.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5419–5429.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Viresh Ranjan, Nikhil Rasiwasia, and CV Jawahar. 2015. Multi-label cross-modal retrieval. In *Proceedings of the IEEE international conference on computer vision*, pages 4094–4102.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR.

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

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3915–3926.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 32:5820–5830.

Jianfei Yu, Luis Marujo, Jing Jiang, Pradeep Karuturi, and William Brendel. 2018. Improving multi-label emotion classification via sentiment classification with dual attention transfer network. In *Proceedings of the 2018 Conference on Empirical Meth-*

ods in Natural Language Processing, pages 1097–1102.

Wenjie Zhang, Junchi Yan, Xiangfeng Wang, and Hongyuan Zha. 2018. Deep extreme multi-label learning. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, pages 100–107.

Fang Zhao, Yongzhen Huang, Liang Wang, and Tieniu Tan. 2015. Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1556–1564.

A Example Appendix

Model	AAPD					EUR-Lex				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
Mix-GCN (G_C)	86.46	62.39	42.88	83.22	87.19	81.83	68.82	56.94	72.23	67.79
Mix-GCN (G_P)	81.56	57.06	37.20	76.66	80.73	79.78	66.12	54.60	70.12	64.34
Mix-GCN (G_F)	**86.98**	**62.56**	**42.97**	**84.02**	**87.43**	**82.11**	**69.02**	**57.22**	**72.68**	**68.01**
Model	RCV1					Kanshan-Cup				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
Mix-GCN (G_C)	97.72	82.39	58.21	94.10	95.85	57.22	35.24	26.69	52.49	54.78
Mix-GCN (G_P)	95.63	80.04	56.26	93.00	94.21	55.01	34.78	25.46	52.68	53.62
Mix-GCN (G_F)	**97.98**	**82.56**	**58.97**	**94.42**	**96.13**	**57.61**	**36.02**	**27.02**	**52.68**	**55.01**

Table 4: Comparisons between state-of-the-art methods on RCV1 and Kanshan-Cup datasets. The bold numbers indicate the best performance. 'G_C', 'G_P', and 'G_F' represent statistical graph, knowledge graph and mixing graph, respectively.

Model	AAPD					EUR-Lex				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
Mix-GCN ($2-L$)	85.54	61.22	41.16	82.70	85.69	80.51	67.78	55.48	71.33	66.64
Mix-GCN ($4-L$)	86.77	62.39	42.45	83.55	86.88	81.88	68.92	56.89	72.46	67.89
Mix-GCN ($3-L$)	**86.98**	**62.56**	**42.97**	**84.02**	**87.43**	**82.11**	**69.02**	**57.22**	**72.68**	**68.01**
Model	RCV1					Kanshan-Cup				
	P@1	P@3	P@5	N@3	N@5	P@1	P@3	P@5	N@3	N@5
Mix-GCN ($2-L$)	96.36	81.20	57.29	93.01	94.73	56.22	34.62	25.67	51.36	53.61
Mix-GCN ($4-L$)	97.59	82.39	58.54	94.29	96.03	57.43	35.88	26.81	52.49	54.92
Mix-GCN ($3-L$)	**97.98**	**82.56**	**58.97**	**94.42**	**96.13**	**57.61**	**36.02**	**27.02**	**52.68**	**55.01**

Table 5: Comparisons between state-of-the-art methods on RCV1 and Kanshan-Cup datasets. The bold numbers indicate the best performance. '$x-L$' indicates the number of layers in Mix-GCN.