Hierarchical Graph Convolutional Network for Multi-Label Text Classification

Bicong Wang †
Harbin Institute of Technology
bcwang@ir.hit.edu.cn

Zekun Wang †
Harbin Institute of Technology
zkwang@ir.hit.edu.cn

Abstract

Multi-**L**abel **C**lassification (MLC) aims to assign multiple labels that are semantically relevant to each instance. However, most previous work neglects modeling the semantic correlation between multiple labels. In this work, we propose a hierarchical graph convolution network (H-GCN), which explicitly models the semantic relationships among labels. Specifically, we take labels, words, and documents as distinct types of nodes in a heterogeneous graph and use the graph network to jointly model the relations in this graph. Experimental results demonstrate that our method improves the baselines on two multi-label classification datasets.

1 Introduction

Multi-**L**abel **C**lassification (MLC) is which has attracted much attention recently. It has been applied in text categorization, information retrieval, tag recommendation and so on. Unlike traditional (binary or multi-class) text classfication, each input instance is assigned with a set of labels. In MLC task, label correlation is known to be a key challenge. For example, the probability of an instance being labeled annotated with label Brazil would be high if we know it has labels rainforest and soccer (Zhang and Zhou, 2014).

Binary relevance (BR) (Boutell et al., 2004) is an early attempt to solve the MLC task, which transform the MLC task into multiple single-label binary classification task. Abviously, it doesn't take advantage of label correlation. Classifier chains (CC) (Read et al., 2011) use a chain of binary classifier to model label correlation. Other adaption methods, including ML-kNN (Zhang and Zhou, 2007), ML-DT (Clare and King, 2001), Rank-SVM (Elisseeff and Weston, 2002), are proposed to solve this problem in machine learning manner. Besides, when neural network models are widely used in natural language processing and text classfication, convolutional neural network (CNN) and recurrent neural network (RNN) are used to capture the semantic information of text (Chen et al., 2017). However, these previous works didn't take the whole graph modeling the relations between labels into consideration.

Recently, graph neural network (GNN) has attracted wide attention. Graph Convolutional Network (GCN) (Kipf and Welling, 2016) is an effective network to model high order neighborhoods information and achieve great success in many areas, including text classification and MLC. TextGCN (Yao et al., 2018) is proposed for single-label text classfication and build a graph between word and document from unstructured text. Also in MLC task, some works (Rios and Kavuluru, 2018; Chen et al., 2019) use GCN to model label correlation. However, the small label graph is weak to learn a better label presentation because of the over-smooth problem in GCN.

In our paper, we propose a hierarchical graph convolution network (H-GCN) for the MLC task. In our model, we build a hierarchical GCN on text, document and label to learn a graph-based feature representation simultaneously with global context, which can benefit from each other in feature extraction. Our contributions are as follows:

- We apply a novel graph neural model for MLC task and proposed a general method to build a heterogeneous graph structure from unstructured texts and labels.
- Our model learn a joint representation for words, documents and labels, which means we model graph-based features between those relations, and this information is extremely important in MLC task.

† Two authors have equal contribution.

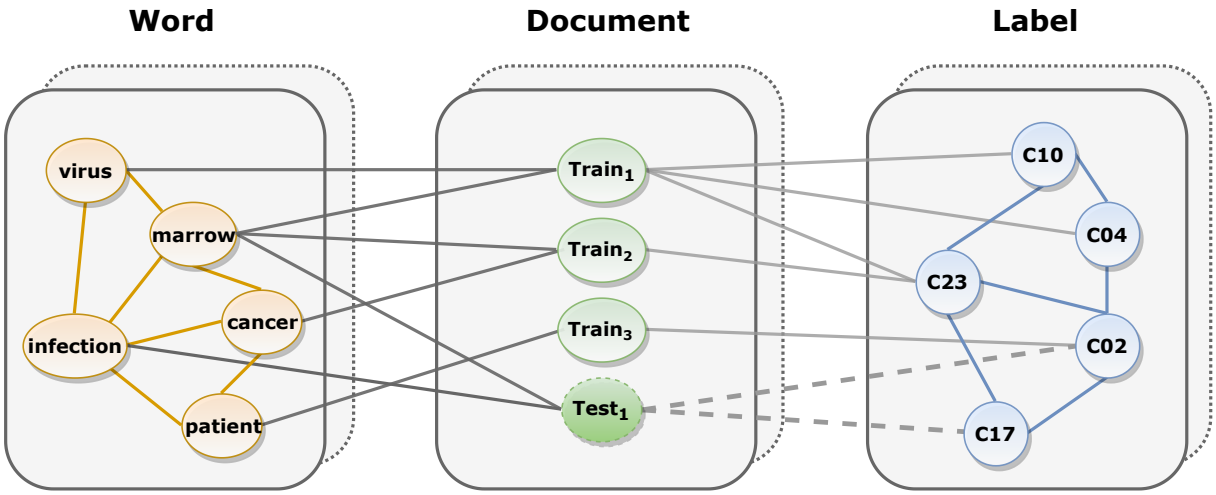

Figure 1: Schematic of H-GCN where examples are taken from Ohsumed dataset. The model have three types of nodes and four types of edges (relations). All the solid line is what we build from unstructured corpus, and the dotted line is what we predict during the evaluation.

2 Method

Figure 1 shows the overall framework of our method. Specifically, we build a heterogeneous graph $G = (V, E)$ that contains word nodes, document nodes, and label nodes. As shown in Figure 1, the edges contain four types of relations. The inner-layer relations (word to word and label to label) and the inter-layer relations (word to document and document to label) make up the heterogeneous structure. Via the inter-layer relations, the second-order relation, such as word to label, can also be defined.

2.1 Graph Structure

Nodes. The graph have three types of nodes, including words, documents and labels in three subgraphs, which formula is $V = V_{word} \cup V_{doc} \cup V_{label}$. The number of word nodes is the number of unique words in the corpus, and it is same to document and label nodes.

Edges. The graph have four types of edges, including:

- Word to word egdes
- Label to label egdes.
- Word to document egdes.
- Document to label egdes.

All the edges in G is $E = E_{w2w} \cup E_{l2l} \cup E_{w2d} \cup E_{d2l}$. The weight of the edges between words is point-wise mutual information (PMI). And we will drop the edges if its PMI isn't more than zero. The PMI value of a word pair i, j is computed as:

$$PMI(i,j) = \log \frac{p(i,j)}{p(i)p(j)} \tag{1}$$

$$p(i,j) = \frac{\#W(i,j)}{\#W} \tag{2}$$

$$p(i) = \frac{\#W(i)}{\#W} \tag{3}$$

Where $\#W(i,j)$ is the number of sliding windows in the overall corpus that contain word i and word j concurrently, $\#W(i)$ is the number of single word i, and W is the number of total sliding windows.

It is same to label to label egdes, the difference is that we use each instance as a sliding window because of disorder.

Moreover, the edges between words and documents use term frequency-inverse document frequency (TF-IDF) as weight and the edges between documents and labels is simply unweighted which means all the existent edges are weighted by 1. The TF-IDF value of a word i and a document j is computed as:

$$TFIDF(i,j) = TF_{i,j} * IDF_i \tag{4}$$

$$TF(i,j) = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{5}$$

$$IDF(i) = \log \frac{|D|}{|\{j : w_i \in d_j\}|} \tag{6}$$

Where $n_{i,j}$ is the number word i in the document j and D is the set of documents in the corpus.

Finally, all the weight between two nodes is defined as:

$$A_{ij} = \begin{cases} PMI(i,j) & i,j \text{ are both words} \\ & i,j \text{ are both labels} \\ TFIDF(i,j) & i \text{ is word, } j \text{ is document} \\ 1 & i \text{ is document, } j \text{ is label} \\ & i = j \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

So we can use the adjacency matrix A as global graph information.

2.2 H-GCN Layer

A graph convolutional network (GCN) is a multi-layer neural network which operates directly on a graph and merges embeddings of nodes based on their neighbor nodes. Each layer is defined as:

$$L^{(j+1)} = GCN(L^{(j)}) = \rho\left(\tilde{A}L^{(j)}W_j\right) \tag{8}$$

Where j denotes the layer number and $L^{(0)}$ denotes the input feature such as one hot vectors. In GCN, we need to build an adjacency matrix A of G and its degree matrix D, where $D_{ii} = \sum_j A_{ij}$. Besides, using a spectral method, we define $\tilde{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$ where I is a unit matrix. ρ is an activation function e.g. a ReLU $\rho(x) = max(0, x)$.

In our model, we proposed a hierarchical GCN (H-GCN) as figure 1 shows, which contains four type of relations corresponding to four origin GCN layers. We define features of words, documents and labels as $\{R_w, R_d, R_l\}$. Obviously, in the first H-GCN layer, R_w^0 is the embeddings for words, and it is same to labels. For documents, we use a pooling layer for inductive learning, which is defined as:

$$R_d^i = Pooling(R_w^i) = \rho\left(\hat{A}R_w^i W_j\right) \tag{9}$$

where $\hat{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$. It's equal to GCN without self-loops which means documents are subgraphs of words and they doesn't have individual features.

For words and labels, we define the message propagation as:

$$R_w^{i+1} = GCN\left(R_w^i\right) \tag{10}$$

$$R_l^{i+1} = \lambda * GCN\left(R_l^i\right) + (1-\lambda) * GCN\left(R_d^i\right) \tag{11}$$

Where λ is a hyperparameter to control the ratio for the two types of relations.

Finally, we use a bilinear function to calculate the matching scores between documents and labels:

$$\hat{y}_{ij} = \sigma\left(R^k_{d_{(i)}} W R^k_{l_{(j)}}\right) \tag{12}$$

2.3 Training

We train our model using a multi-label binary cross-entropy loss (Nam et al., 2014) as follows:

$$\mathcal{L} = \sum_{i=1}^{|D|}\sum_{j=1}^{|L|}\left[-y_{ij}\log\left(\hat{y}_{ij}\right) - (1-y_{ij})\log\left(1-\hat{y}_{ij}\right)\right] \tag{13}$$

where y is the ground truth label of the instance in the corpus and $\hat{y}$ is the matching score logits between documents and labels.

3 Experiments

3.1 Datasets and Evaluation

Ohsumed (Hersh et al., 1994). The Ohsumed test collection is a subset of the MEDLINE database, which is a bibliographic database of important, peer-reviewed medical literature maintained by the National Library of Medicine. After selecting such category subset, the document number is 13,929 (6,286 for training, 3,822 for valuation and 3,821 for testing in our split). An instance may have multiple subjects and there are 23 subjects in total.

AAPD (Yang et al., 2018). Arxiv Academic Paper Dataset is a large dataset for the multi-label text classification. It is collected from the abstract and the corresponding subjects of 55,840 papers in the computer science field from the arxiv. An academic paper may have multiple subjects and there are 54 subjects in total.

The summary statistics of the datasets are shown in Table 1. Following the previous work, we choose Micro F1 as the evaluation metric. Micro F1 is calculated globally by count the total true positives (TP), false negatives (FN), and false positives (FP).

Dataset	Train	Valuation	Test	Labels	Doc length	Vocabulary
Ohsumed	6286	3822	3821	23	129.12	15518
AAPD	53840	1000	1000	54	101.81	23814

Table 1: Summary statistics of Ohsumed and AAPD datasets.

Model	Ohsumed	AAPD
TF-IDF+SVM	0.588	0.668
SGM (Yang et al., 2018)	0.493	**0.676**
TextGCN(Yao et al., 2018)	0.617	0.663
Our model	**0.629**	0.658

Table 2: Micro F1 on Ohsumed and AAPD datasets between baselines and our model.

Model	Micro F1
TextGCN	0.617
TextGCN + $L2L$	0.616
TextGCN + $L2L$ + $D2L$	0.616
Our model	**0.629**

Table 3: Ablation experiments. Micro F1 on Ohsumed using our model. $L2L$ means label to label edges and $D2L$ means document to label edges. The difference between our model and TextGCN* is that our model use H-GCN layer but TextGCN use GCN layer.

3.2 Settings

For both datasets, we lowercase all words in documents and labels. we remove the stop words and low-frequency words which count is less than 5. The vocabulary is extracted from training sets where out-of-vocabulary (OOV) words are replaced with unk. We use 300-dimensional GloVe (Pennington et al., 2014) as the pre-trained word embedding. We set the window size as 20 for PMI to build the graph. The λ in Equation 11 as 0.5. We train our model with Adam (Kingma and Ba, 2014) optimizer within 1000 epochs. The learning rate is set as 0.1 and the dropout rate is set to 0.5.

3.3 Baselines

SGM (Yang et al., 2018) is a sequence-to-sequence model for multi-label classification. TextGCN (Yao et al., 2018) uses GCN for single-label text classfication, and we use binary relevance (BR) strategy for test.

3.4 Results

Test Performance. Table 2 presents the evaluation results for each model. Our model outperforms the baselines in the Ohsumed dataset and achieves a comparable result in the AAPD dataset.

Effects of the label to label and document to label edges.. In order to evaluate the effect of label to label and document to label edges, we tested several strategies using TextGCN with different graph structure. The difference between our model and TextGCN* is that our model use H-GCN layer but TextGCN use GCN layer, which means TextGCN take the heterogeneous graph structure into a single GCN, but our model use hierarchical GCN to learn four subgraphs for multi-relations. By the experiments shown in Table 3, we found that a complex edges (relations) is difficult for GCN to achieve a better result because of heterogeneity. However, by hierarchical multi GCNs, our model can learn a better graph representation.

4 Conclusion

In this study, we propose the hierarchical graph convolutional network (H-GCN) for multi-label text classification task. We first build a heterogeneous graph for words, documents and labels, and then design a hierarchical GCN to model the multi relations between these. This model can help us to learn a better representation for both labels and words via correlation. The experimental results show that our method improves the baselines on two multi-label classification datasets.

References

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

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.

Amanda Clare and Ross D King. 2001. Knowledge discovery in multi-label phenotype data. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 42–53. Springer.

André Elisseeff and Jason Weston. 2002. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pages 681–687.

William Hersh, Chris Buckley, T. J. Leone, and David Hickam. 1994. Ohsumed: An interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 192–201, New York, NY, USA. Springer-Verlag New York, Inc.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification — revisiting neural networks. *Lecture Notes in Computer Science*, page 437–452.

Jeffrey Pennington, Richard Socher, and Christopher 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.

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

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 3132. NIH Public Access.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: Sequence generation model for multi-label classification.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2018. Graph convolutional networks for text classification.

Min-Ling Zhang and Zhi-Hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048.

Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.