Sparse Structure Learning via Graph Neural Networks for Inductive Document Classification

Yinhua Piao,1Sangseon Lee,2Dohoon Lee,3Sun Kim1,3,4,5
1Department of Computer Science and Engineering, Seoul National University
2Institute of Computer Technology, Seoul National University
3Bioinformatics Institute, Seoul National University,4AIGENDRUG Co., Ltd.
5Interdisciplinary Program in Artificial Intelligence, Seoul National University

AAAI2022

Code:https://github.com/qkrdmsghk/TextSSL

2022.4.19 • ChongQing

Reported by Yang Peng

Artificial

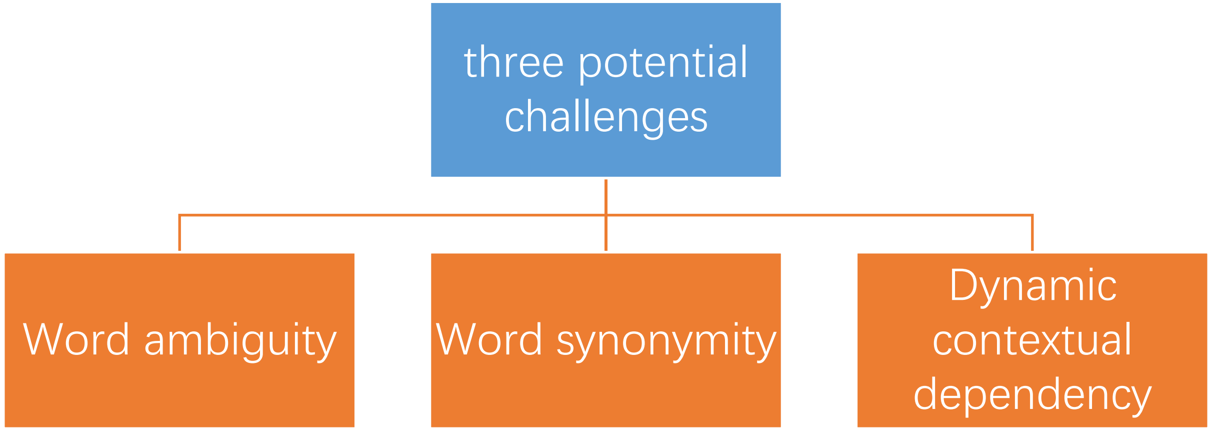
1.Introduction

2.Method

3.Experiments

Introduction

- Document classification, a task of using algorithms to automatically classify the input document to one or multiple categories. Nevertheless, almost all graph-based methods are de-signed to construct static word co-occurrence graph for the whole document without considering sentence-level information.



- We construct a trainable individual graph consisting of sentence-level subgraphs for each document.
- We propose a sparse structure learning model via GNNs to learn an effective and efficient structure with dynamic syntactic and semantic information for each document.

Method

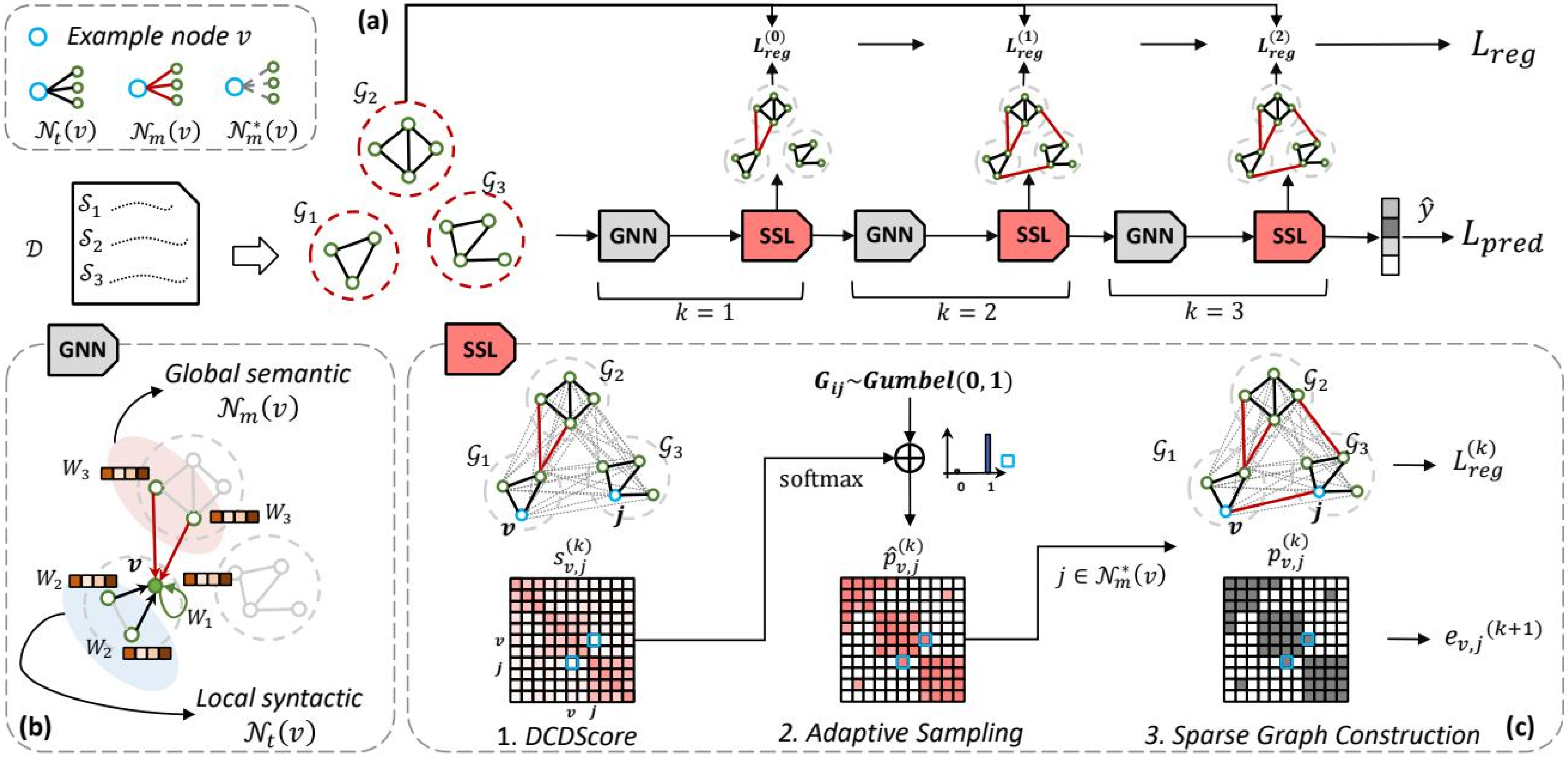
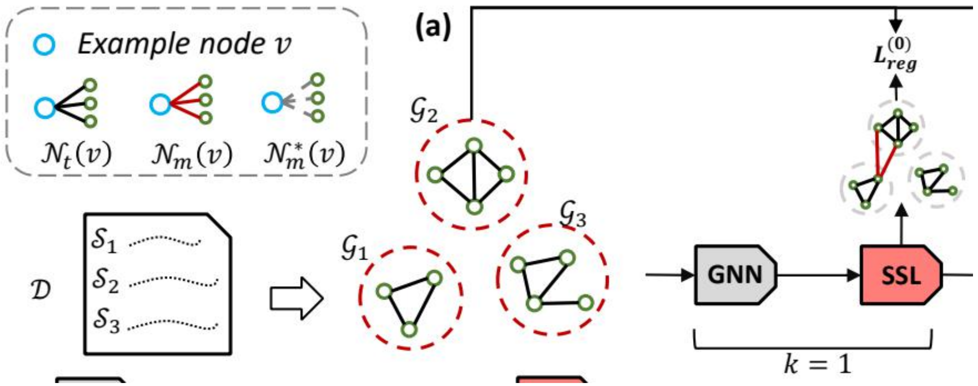
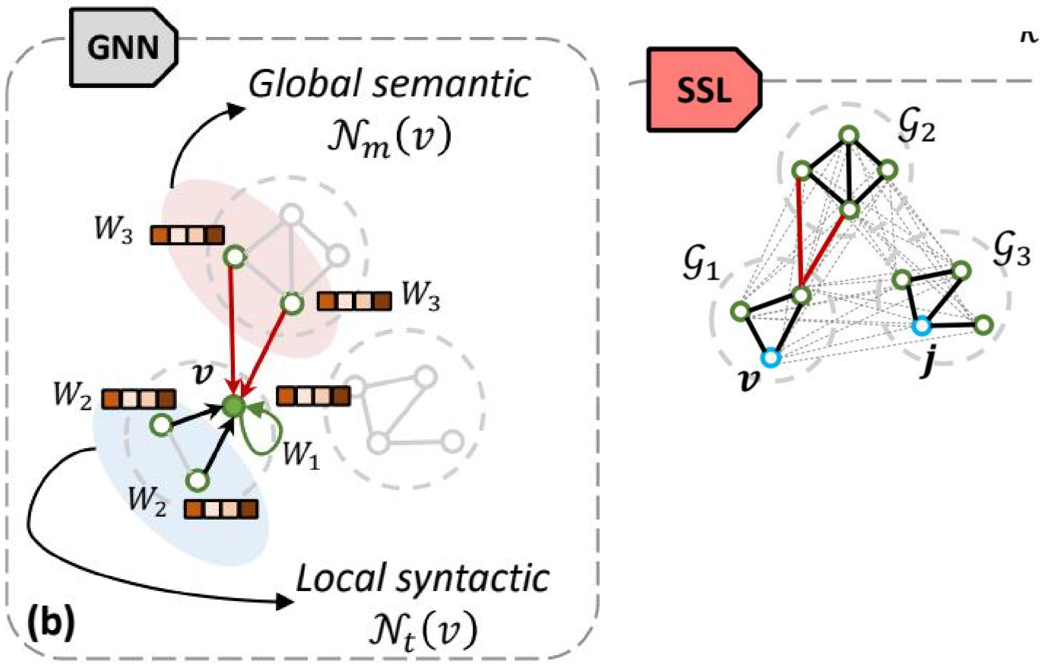


Figure 1: Overview of the proposed model. (a) Model framework. (b) GNN: Local and Global Joint Message Passing. (c) SSL: Sparse Structure Learning contains (c.1) Dynamic Contextual Dependency Score, (c.2) Adaptive Sampling for Sparse Structure, and (c.3) Reconstructing Sparse Graph.





Method

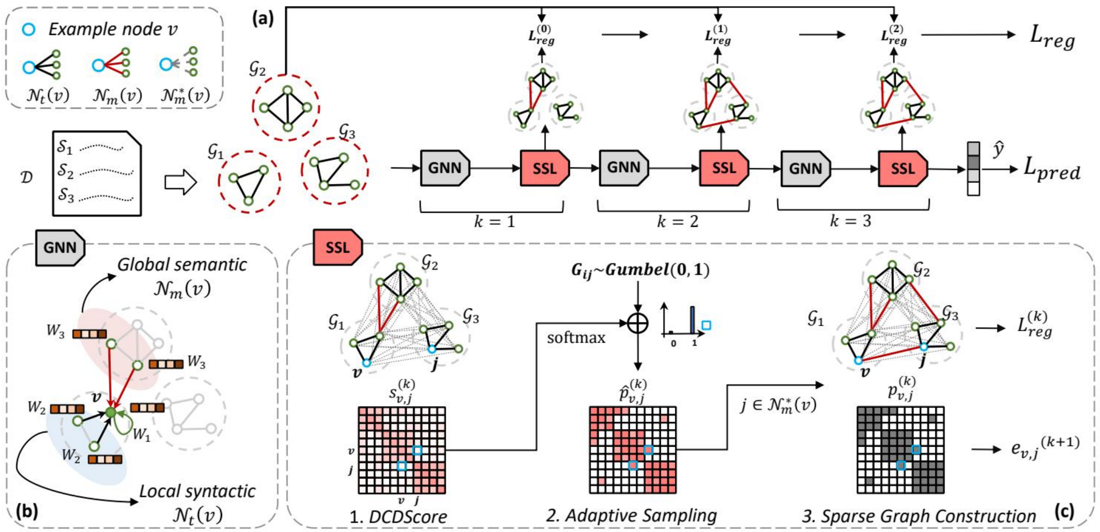
Graph Construction

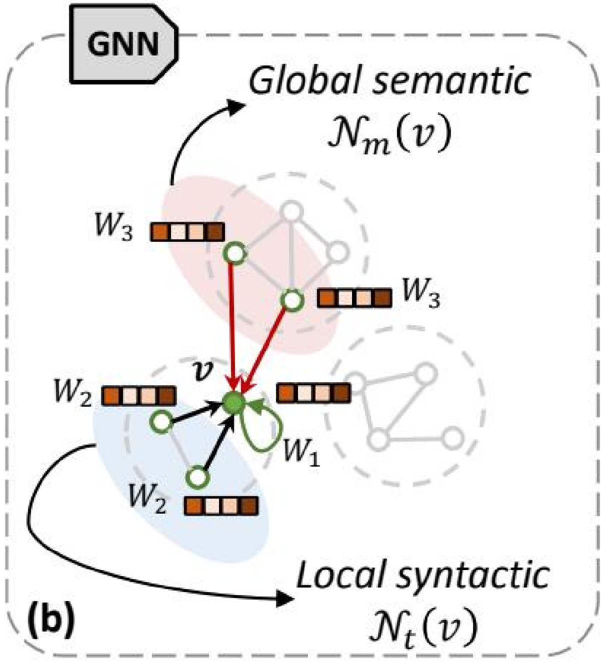
Definition 1. Sentence-level Subgraph Given a sentence $s_i \in \mathcal{S}$, a sentence-level subgraph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ can represent the sentence s_i as a word co-occurrence graph. The node set $\mathcal{V}_i$ contains words in sentence s_i. The edge set $\mathcal{E}_i$ contains all connections between any pair of words in $\mathcal{V}_i$

Definition 2. Local Syntactic Neighbor Given a node $v \in \mathcal{V}$ in a preliminary document graph $\tilde{\mathcal{G}}$, we define a local syntactic neighbor $u \in \mathcal{N}_t(v)$ that is adjacent to node v within sentence-level subgraphs $\mathcal{G}_\mathcal{S}$.

Definition 3. Global Semantic Neighbor Given a node $v \in \mathcal{V}$ in a preliminary document graph $\tilde{\mathcal{G}}$, we define a global semantic neighbor $z \in \mathcal{N}_m(v)$ that can have dynamic relation with node v between sentence-level subgraphs $\mathcal{G}_s$.

A document-level graph $\mathcal{G} = (\mathcal{V}, \{\mathcal{E}_t \cup \mathcal{E}_m\})$





Method

Local and Global Joint Message Passing

k-th iteration of message passing process in a GNN

$$h_v^k = \phi\left(f^{(k)}(h_v^{(k-1)}, \{h_u^{(k-1)} : u \in \mathcal{N}_v\})\right), \qquad (1)$$

obtain the entire graph's representation

$$h_G = R(\{h_v^{(K)} | v \in G\}). \qquad (2)$$

The message passing part can be reformulated as:

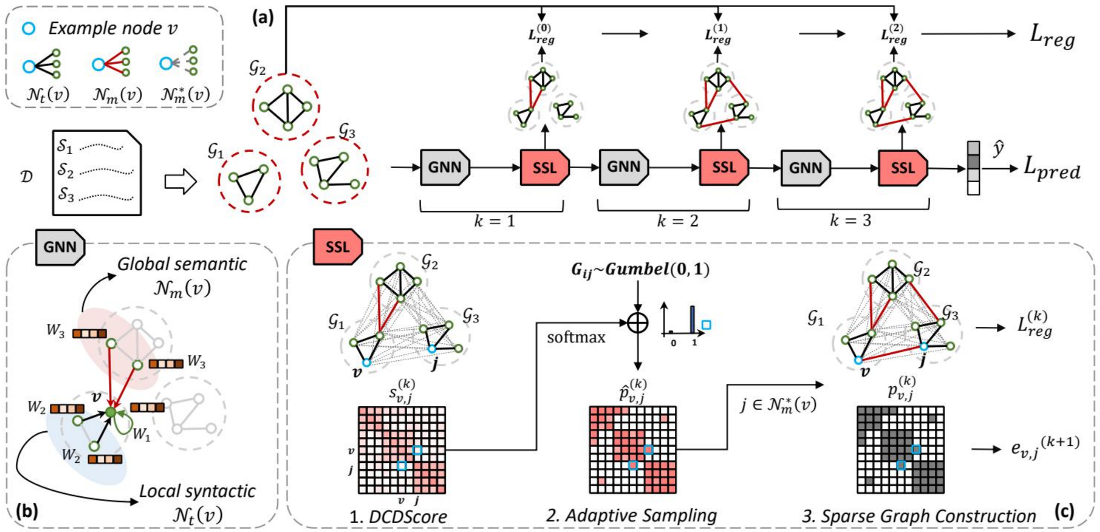
$$h_v^{(k)} = \phi\left(h_v^{(k-1)}\mathbf{W}_1^{(k)} + t_v^{(k)}\mathbf{W}_2^{(k)} + m_v^{(k)}\mathbf{W}_3^{(k)}\right), \qquad (5)$$

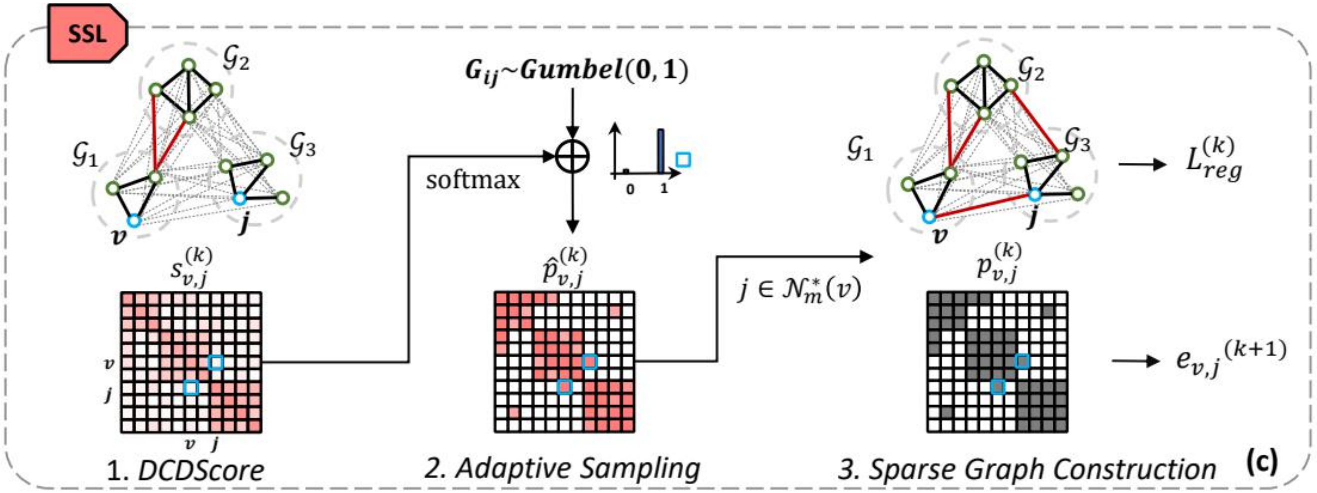
$h_v^{(k)} \in \mathbb{R}^b$ is the node representation vector and b is the number of hidden dimension. The local syntactic neighbor representations $t_v^{(k)} \in \mathbb{R}^b$ and global semantic neighbor representations $m_v^{(k)} \in \mathbb{R}^b$ can be expressed as:

$$t_v^{(k)} = \sum_{u \in \mathcal{N}_t(v) \cup \{v\}} \frac{e_{u,v}}{\sqrt{\hat{\zeta}_u \hat{\zeta}_v}} h_u^{(k-1)} \qquad (6)$$

$$m_v^{(k)} = \sum_{z \in \mathcal{N}_m(v)^{(k-1)}} \frac{e_{z,v}}{\sqrt{\hat{\zeta}_z \hat{\zeta}_v}} h_z^{(k-1)} \qquad (7)$$

$\hat{\zeta}_v = \sum_{j \in \mathcal{N}} \hat{A}_{vj}$ with self-looped adjacency matrix $\hat{A} = A + I$.





Method

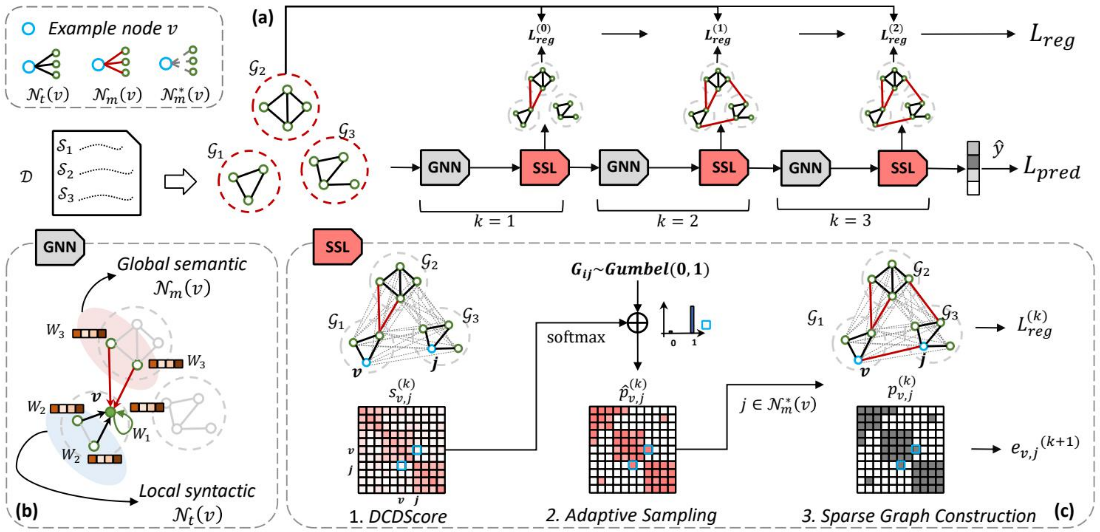
Sparse Structure Learning

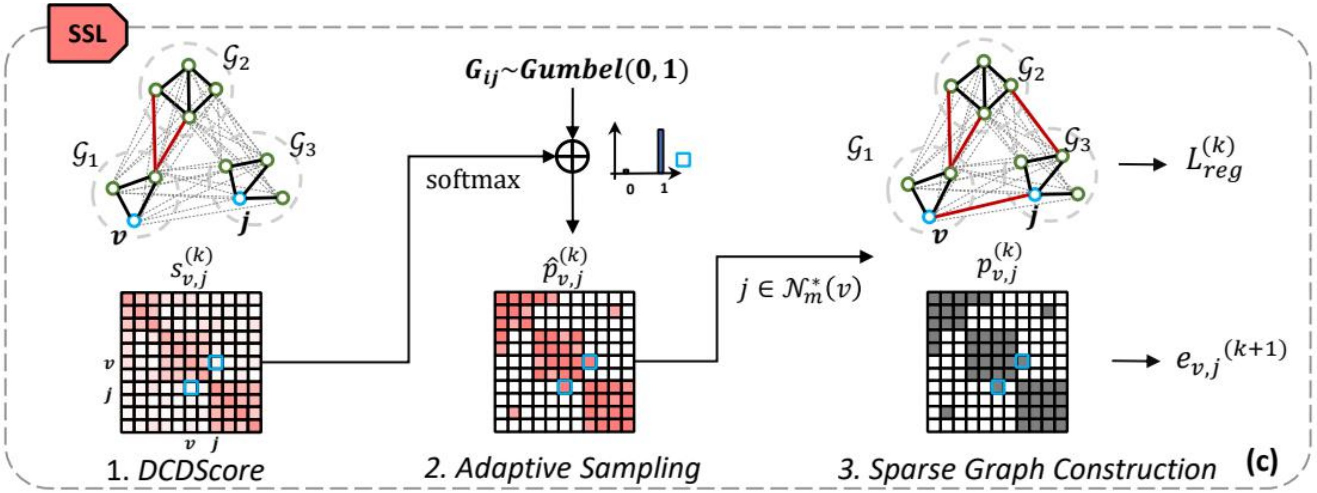
Dynamic Contextual Dependency Score Given a node $v \in \mathcal{V}$ in a complete graph $\mathcal{G}^*$, all neighbors of node v are in $\mathcal{N}^*(v)$, where we can obtain $\mathcal{N}_m^*(v) = \mathcal{N}^*(v) - \mathcal{N}(v)^{(k-1)}$ that contains all global semantic candidate neighbors of node v. We first calculate *attention coefficient score* between each neighbor $j \in \mathcal{N}^*(v)$ and node v as follows:

$$a_{v,j}^{*(k)} = \psi\left(\mathbf{a}^{(k)\top}[h_v^{(k)}\mathbf{W}^{(k)} || h_j^{(k)}\mathbf{W}^{(k)}]\right) \qquad (8)$$

where $\mathbf{W}^{(k)} \in \mathbb{R}^{b\times b}$ denotes the projection for node features $h_v \in \mathbb{R}^{1\times b}$ and $h_j \in \mathbb{R}^{n\times b}$. k denotes the current layer of our model. We adopt function ψ as LeakyReLU$(\cdot)$ activation function, and $\mathbf{a} \in \mathbb{R}^{b\times 1}$ is a learnable vector.

$$s_{v,j}^{(k)} = \frac{\exp(a_{v,j}^{*(k)})}{\sum_{u\in\mathcal{N}^*(v)}\exp(a_{v,u}^{*(k)})}. \qquad (9)$$





Method

Sparse Structure Learning

Gumbel-Softmax Distribution

Formally, let a discrete variable π has a distribution of probabilities $(\phi_1, ..., \phi_n)$ with class $C = \{c_1, ..., c_n\}$. Gumbel-max (Gumbel 1954) provides an efficient way for the categorical distribution to sample x_π with:

$$x_\pi = \operatorname{argmax}(\log \phi_i + G_i) \qquad (3)$$

Gumbel-Softmax to approximate it as follows:

$$\hat{x}_\pi = \frac{\exp((\log(\phi_i) + G_i)/\tau)}{\sum_{j=1}^{n} \exp((\log(\phi_j) + G_j)/\tau)} \qquad (4)$$

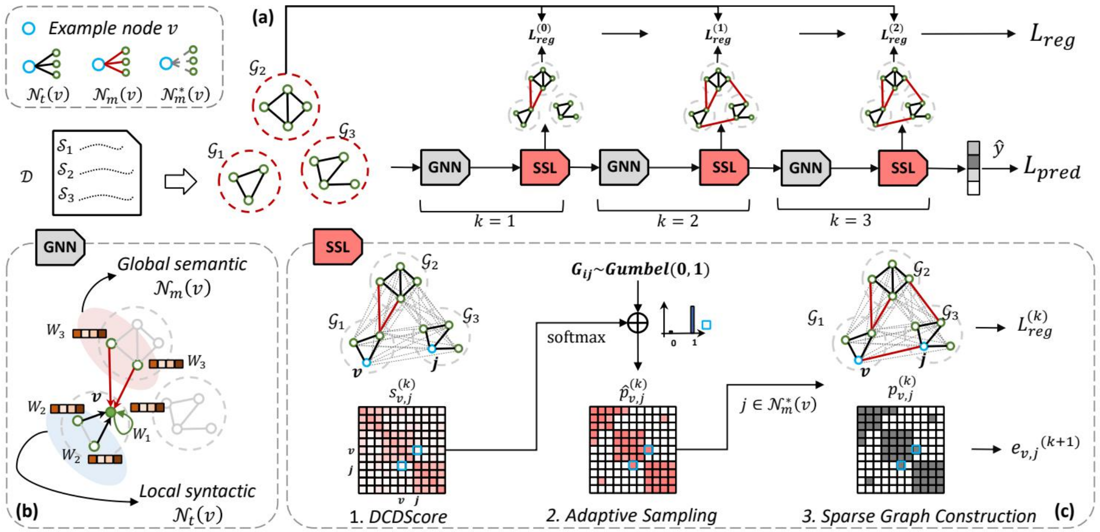
Sampling Adaptive Neighbors for Sparse Structure

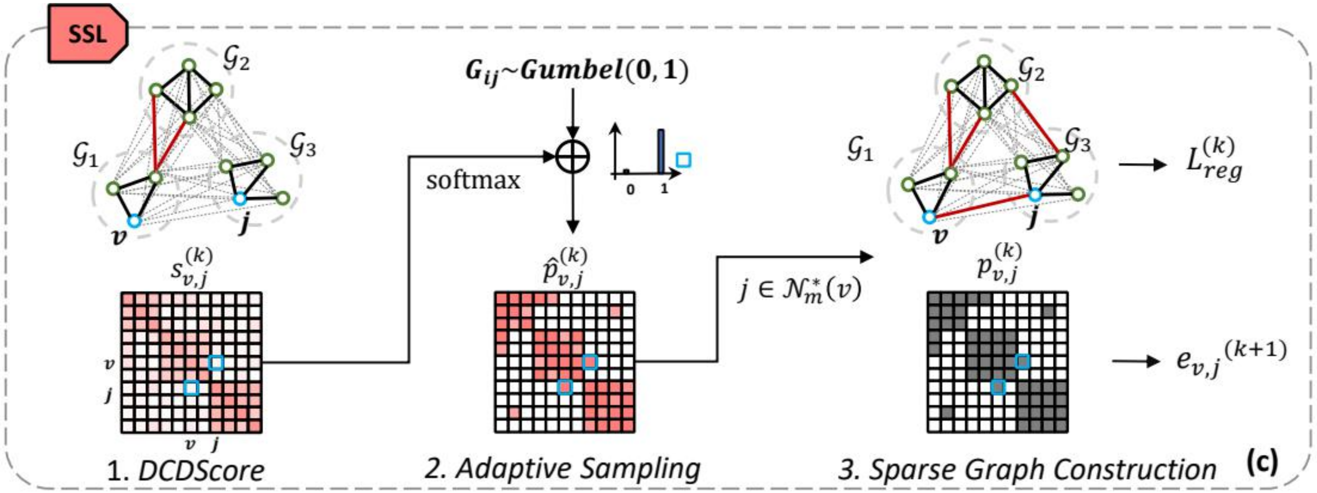
$\{\pi_1 := s_{v,j}^{(k)}, \pi_0 := 1 - s_{v,j}^{(k)}\}$ and adopt Gumbel-Softmax approach to generate differentiable probability $\hat{p}_{v,j}^{(k)}$ of selector samples $p_{v,j}^{(k)}$ as follows:

$$\hat{p}_{v,j}^{(k)} = \frac{\exp((\log \pi_1 + g_1)/\tau)}{\sum_{i \in \{0,1\}} \exp((\log \pi_i + g_i)/\tau)}, \qquad (10)$$

Method

Reconstructing Sparse Graph





document graph. Specifically, we update the global semantic neighbors $\mathcal{N}_m(v)^{(k)}$ for node v with selected candidate neighbors as follows:

$$\mathcal{N}_m(v)^{(k)} = \mathcal{N}_m(v)^{(k-1)} \cup \{j \,||\, \forall j \to p_{v,j}^{(k)} = 1\}. \quad (11)$$

where $j \in \mathcal{N}_m^*(v)$. In addition, for static local syntactic neighbors $\mathcal{N}_t(v)$, we compute the entropy to preserve consistency of the original syntactic information and prevent too much structure variation in the graph.

$$L_{reg}^{(k)} = \sum_{v\in\mathcal{V}} \sum_{j\in\mathcal{N}_t(v)} -\hat{p}_{v,j}^{(k)} \log(\hat{p}_{v,j}^{(k)}), \quad (12)$$

$$L_{pred} = l(R(h_v), y), \quad (13)$$

Experiments

Dataset	#Docs	#Training	#Test	#Classes (ρ)	#Vocab.	Avg.#Length	Avg.#Sentence	#Prop.NW
MR	10,662	7,108	3,554	2 (1.0)	18,764	20.39	1.17	30.09%
R8	7,674	5,485	2,189	8 (84.7)	7,688	65.72	4.03	2.60%
R52	9,100	6,532	2,568	52 (1666.7)	8,892	69.82	4.34	2.63%
Ohsumed	7,400	3,357	4,034	23 (62.5)	14,157	135.82	8.59	8.46%
20NG	18,846	11,314	7,532	20 (1.6)	42,757	221.26	6.06	7.40%

Table 1: Statistics of the datasets. ρ denotes class imbalance ratio (the sample size of the most frequent class divided by that of the least frequent class). The Avg.#Length and the Avg.#Sentence mean the number of words and the number of sentences in a document, respectively. The #Prop.NW denotes the proportion of new words in test.

Experiments

Categories	Baselines	MR	R8	R52	Ohsumed	20NG
Word-based	fastText	72.17±1.30	86.04±0.24	71.55±0.42	14.59±0.00	11.38±1.18
	SWEN	76.65±0.63	95.32±0.26	92.94±0.24	63.12±0.55	85.16±0.29
Sentence-based	CNN-non-static	77.75±0.72	95.71±0.52	87.59±0.48	58.44±1.06	82.15±0.52
	LSTM (pretrain)	77.33±0.89	96.09±0.19	90.48±0.86	51.10±1.50	75.43±1.72
	Bi-LSTM	77.68±0.86	96.31±0.33	90.54±0.91	49.27±1.07	73.18±1.85
Graph-based (Tr)	TextGCN	76.74±0.20	97.07±0.10	93.56±0.18	68.36±0.56	86.34±0.09
	Huang et al.	-	97.80±0.20	94.60±0.30	69.40±0.60	-
	TensorGCN	77.91±0.07	**98.04±0.08**	95.05±0.11	70.11±0.24	**87.74±0.05**
	DHTG	77.21±0.11	97.33±0.06	93.93±0.10	68.80±0.33	87.13±0.07
Graph-based (Ind)	TextING	78.93±0.65	97.34±0.25	93.73±0.47	67.95±0.52	OOM
	HyperGAT	77.36±0.22	96.82±0.21	94.15±0.18	66.39±0.65	84.65±0.31
	Our proposal	**79.74±0.19**	97.81±0.14	**95.48±0.26**	**70.59±0.38**	85.26±0.28

Table 2: Test accuracies of various models on five benchmark datasets. The mean ± standard deviation of all models are reported an average of 10 executions of each model. Graph-based (Tr) means transductive graph-based methods and Graph-based (Ind) means inductive graph-based methods.

Experiments

Graph	R8	R52	Ohsumed
WordCooc	97.20±0.29	93.82±0.15	68.08±0.32
Disjoint	97.29±0.21	94.80±0.20	69.72±0.27
Complete	97.40±0.25	94.35±0.10	67.57±0.30
Ours	97.76±0.16	95.32±0.21	70.53±0.30
Ours w/ reg	**97.81±0.14**	**95.48±0.26**	**70.59±0.38**

Table 3: Comparison with different constructions of document-level graphs. (1) WordCooc denotes word co-occurrence graph. (2) Disjoint means a disjoint union of sentence-level subgraphs. (3) Complete graph means disjoint graph with fully connected edges between sentences. (4) Ours graph is constructed by sentence-level subgraphs and learned by sparse structure learning(w/ reg means we add regularization to our model).

τ	R8	R52	Ohsumed
0.01	97.50±0.29	95.16±0.18	**70.59±0.38**
0.1	97.34±0.13	**95.48±0.26**	70.21±0.40
0.2	97.44±0.39	95.03±0.16	70.33±0.32
0.5	**97.81±0.14**	94.56±0.33	70.34±0.37
1.0	97.35±0.24	95.09±0.32	70.22±0.29

Table 4: Test accuracy with different temperatures τ for adaptive sampling.

Experiments

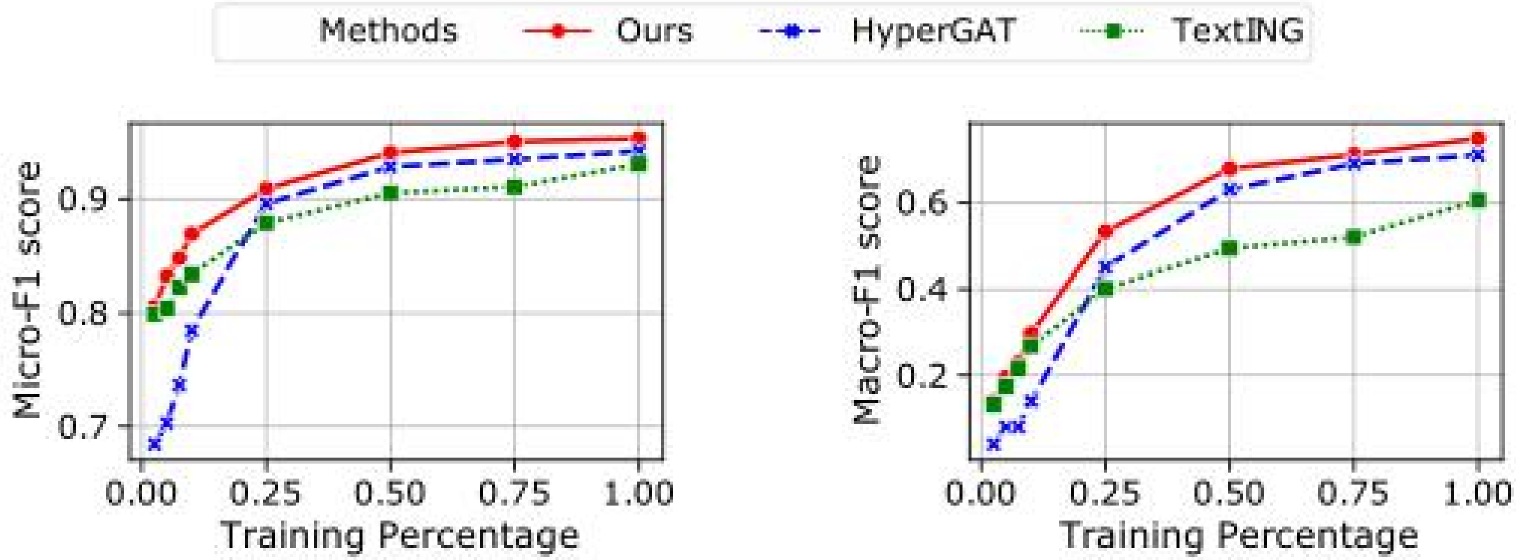


Figure 2: Micro F1 score and Macro F1 score with different percent of training data from 0.025 to 1 on R52 dataset.

Thank you!