Interactive Lexical and Semantic Graphs for Semisupervised Relation Extraction

Wanli Li, *Member, IEEE*, Tieyun Qian, *Member, IEEE*, Ming Zhong, and Xu Chen

***Abstract*— The performance of relation extraction (RE) is hindered by the lack of sufficient labeled data. Semisupervised methods can offer to help hands with this problem by augmenting high-quality unlabeled samples into the training data. However, existing semisupervised RE methods either need a set of manually defined rules or rely on the classifier trained on the small labeled data, i.e., the former requires the heavy intervention of human knowledge, and the latter is bound to the number and the quality of the labeled data. In this article, we present a novel semisupervised RE method that involves small human efforts and is robust to the size of the initial set of labeled data. Specifically, we adopt only two simple rules to build the lexical and semantic graphs which connect the labeled samples with the unlabeled ones. In this way, the graphs are much easier to construct yet keep the ability to transfer knowledge from labeled samples to unlabeled ones. We then develop a graph interaction module to fully exploit the reference information in lexical and semantic graphs, which is used to jointly recognize the high-quality unlabeled samples with the classifier. We conduct extensive experimental results on two public datasets. The results demonstrate that our proposed method significantly outperforms the state-of-the-art baselines.**

***Index Terms*— Lexical and semantic graphs, prior knowledge, relation extraction (RE), semisupervised learning.**

I. Introduction

RELATION extraction (RE) aims to predict the relation between the entities mentioned in the text. RE tasks can be categorized into sentence-level [1], [2] and document-level types [3], where two entities belong to one sentence or multiple sentences, respectively. In this article, we focus on the sentence-level RE task. For example, given a sentence "*The acceptance mail is sent to the author*," there are two entities *"e1: mail"* and *"e2: author,"* and the goal is to predict the relation between these two entities, i.e., *"entity:destination (e1, e2)*."

RE has long been recognized as an important task in natural language processing and has aroused much research attention in recent years. RE facilitates transforming massive, unstructured text into structured factual knowledge. Consequently, it has a wide range of downstream applications [4]–[6], including knowledge graph construction [7], biomedical knowledge discovery [8], question answering [9], and information retrieval [10].

Early studies in RE mostly concentrate on two types of methods: supervised learning [2], [11]–[14] and distant supervision (DS) [15]–[17] methods. Supervised learning often requires a large number of labeled data to train a good classifier [18]. However, the acquisition of reliable and high-quality labeled data has been severely limited by the time and expense of the manual annotation process. DS employs the knowledge bases (KBs) to automatically annotate two entities co-occurring in one sentence with their KB relation. The assumption is that if two entities belong to a certain relation in KB, then all sentences mentioning these entities indicate this relation. While DS methods can tackle the problem of manual labeling, they inevitably generate false-positive and false-negative samples. Moreover, it might be hard to apply DS to RE because the target relations cannot be found in existing KBs in some cases.

The semisupervised learning methods [19], [20] can offer to help hands in alleviating the scarcity problem of labeled data in RE. The essence of semisupervised RE lies in iteratively training a classifier that predicts the unlabeled samples and then chooses the high-quality ones to augment the labeled data. The augmented samples play a key role in promoting the model's generalization ability in semisupervised scenarios. Most of the existing studies along with this line use rule-based approaches under the bootstrapping framework [21]–[23] to select high-quality samples. However, the rules still need to be carefully defined with the heavy intervention of human knowledge. Self-training [24] only has one module; thus, it is unable to deal with the generated bias. Moreover, mean-teacher [25] has two similar modules, and the augmented data will have a similar bias. Consequently, both these methods are not suitable for RE since they are unable to address the semantic drift problem [26]. A recent work [18] presents a novel semisupervised and rule-free approach DualRE. The basic idea is to introduce an additional retrieval module besides the original relation prediction module (classifier) such that two modules can collaborate with each other and improve the quality of annotation. Thanks to the power of joint learning, DualRE has achieved remarkable improvements over the semisupervised methods in several public RE benchmarks.

Manuscript received 12 May 2021; revised 5 October 2021 and 8 December 2021; accepted 25 December 2021. Date of publication 10 January 2022; date of current version 6 October 2023. This work was supported in part by NSFC under Project 61572376, Project 62032016, and Project 61972291; and in part by the State Grid Technology under Project 5700-202072180A-0-00-00. *(Corresponding author: Tieyun Qian.)*

The authors are with the School of Computer Science, Wuhan University, Wuhan 430072, China (e-mail: wanli.li@whu.edu.cn; qty@whu.edu.cn; clock@whu.edu.cn; xuchen@whu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3138956.

Digital Object Identifier 10.1109/TNNLS.2021.3138956

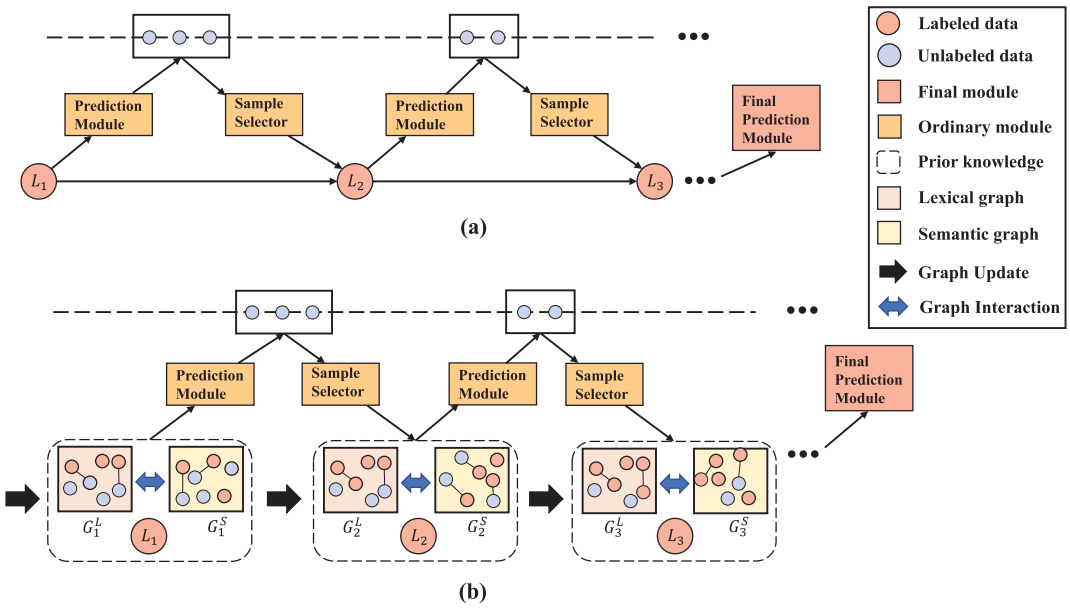


Fig. 1. Illustration of the difference between (a) existing semisupervised relation extraction model and (b) proposed semisupervised relation extraction model.

While being effective, DualRE suffers from one severe problem, i.e., the quality of augmented data highly depends on the mapping function $\boldsymbol{y} = f(\boldsymbol{x})$, where $\boldsymbol{x}$ is the feature vector of the instance in unlabeled data and $\boldsymbol{y}$ is the relation of the entities in the sample. Note that each sample is transformed into the feature vector using an encoder and the function $f()$ is learned using the training corpus. In other words, only if the sample $\boldsymbol{x_u}$ in the unlabeled data is close to the sample $\boldsymbol{x_t}$ in the training data in the latent vector space, can the function $f()$ assign the entities in x_u to the same relation y of the entities in x_t. However, the position in which an instance is located is determined by all components in the instance, and thus, it is easily affected by various modifiers. For example, the unlabeled sample "It was first applied to an eating *establishment* in around 1765 founded by a Parisian *soup-seller* named Boulanger." is far away from the labeled sample "*Steve Jobs* founded *Apple*." Consequently, it is hard for the mapping function to assign the correct "*org:founded_by* (*establishment*, *soup-seller*)" relation to the unlabeled sample.

The above-mentioned observations motivate this study: can we leverage the rule-based method with small human efforts and benefit from the joint learning framework at the same time? To this end, we propose a lexical and semantic graph interaction model which only involves two rules and let it jointly train with the classifier. Specifically, we construct the lexical graph using the rule of "same entity or verb" and the semantic graph using the rule of "similar representations." In this way, we build the connection between the labeled and unlabeled data with only two simple rules. We then design a graph interaction module to deeply fuse the information from these two types of graphs. We also let the graph interaction module collaborate with the classifier to make the prediction on the unlabeled data more reliable. Finally, the recognized high-quality samples are used to augment the training data.

Fig. 1 gives an illustration of the difference between existing semisupervised methods and our proposed model. It can be seen that we extend the RE process to a more accurate one with the help of constructed lexical and semantic graphs. Our preliminary study [27] connects the labeled and unlabeled data using the lexical graphs if two sentences have the same entities or same verbs. This involves less human intervention, whereas the semantic relations between labeled and unlabeled samples have not been explored yet, not to mention the interaction between lexical and semantic graphs.

To summarize, we make the following contributions.

1) We propose to combine the rule-based method and the joint learning framework for better recognizing high-quality unlabeled samples for semisupervised RE.
2) We present two simple rules to construct the lexical and semantic graphs and design a joint learning neural network to let them collaborate with the prediction module.
3) We conduct extensive experiments on two public datasets. Results prove that our method outperforms the state-of-the-art supervised and semisupervised baselines.

II. Related Work

In this section, we review the literature in relation exaction and graph neural networks (GNNs).

A. Relation Extraction

The majority of early RE models utilize traditional machine learning algorithms, such as kernel methods [11], hidden Markov models (HMMs) [28], maximum entropy Markov models [29], and conditional random fields [30]. More recently, with the success of deep learning techniques, a good number of neural models are proposed for RE tasks, including recurrent neural network (RNN) [31], convolutional neural network (CNN) [32], bidirectional long short-term memory (Bi-LSTM) [33], piecewise CNN (PCNN) [14], and position-aware RNN (PRNN) [2].

The aforementioned methods are all supervised ones and require a large number of labeled data to train an effective classifier. To address this problem, there are mainly two types of approaches in RE. One is DS learning, which relies on the KBs to obtain label information [15]–[17]. The other is semisupervised learning [19], [21], [23], which is more general than DS learning in the sense that it does not need external KB resources. However, most of the semisupervised learning approaches are rule-based [21]–[23] and still involve manual operations. A recent semisupervised method DualRE [18] is proposed to automatically select unlabeled samples by using two collaborative modules. The main advantage of DualRE is that it does not need predefined or annotated rules. However, the performance of DualRE is limited by the initial set of labeled data since both the retrieval and prediction modules are dependent on the labeled set.

Compared with the rule-based methods, our lexical and semantic graph interaction (LSGI) model is much simpler as it only involves two essential rules. On the one hand, LSGI takes advantage of DualRE [18], in which we also adopt the joint learning framework where different modules can help each other. On the other hand, LSGI is different from DualRE, in which LSGI exploits the connection between unlabeled instances and labeled ones in an unsupervised way, rather than directly building a mapping function in DualRE. The performance in DualRE is extremely affected by the amount of training data since its retrieval module selects sentences expressing a relation in a supervised way. In other words, without sufficient training data, it is hard for DualRE to learn a good mapping function. In contrast, by leveraging the lexical and semantic reference information, our LSGI can capture the prior knowledge to connect the labeled and unlabeled data, and thus, it is more robust and can achieve better performance than DualRE when the ratio of labeled data is small, as we will show in our experiments. Finally, the difference between LSGI and our previous MRefG method [27] lies in that MRefG contains neither the semantic graph nor the graph interaction module. Consequently, the performance of MRefG is much inferior to LSGI.

B. Graph Neural Networks

Our model is related to GNN [34]–[36] since we propagate information in the constructed lexical and semantic graphs. The concept of GNN is first proposed in [37], which extends existing neural network for processing the graph data. GNN aims to learn for a target node v the representation that contains v's neighborhood information. Recently, some researchers propose to apply the self-attention strategy into the propagation step whose great potential has been well demonstrated in various areas. Heterogeneous graph attention network (HAN) [38] further applies the above-mentioned mechanism to heterogeneous graphs and develops a HAN.

GNN has been widely adopted in various graph analysis tasks due to its convincing performance and high interpretability. Several recent attempts in RE have already taken the advantage of the GNN into account. For example, attention guided graph convolutional networks (AGGCN) [13] takes full dependency trees conveying rich structural and syntactic information as inputs of the graph. GraphRel [39] uses the dependency tree as the sentence's adjacency matrix and adopts GCN in addition to LSTM to extract features. A walk-based model [40] constructs entity-based graphs to encode the dependencies between relations.

All the above-mentioned methods are built upon the intrasentence graphs where two nodes belong to the same sentence. To the best of our knowledge, we are the first that exploit the intersentence graphs to connect the labeled and unlabeled data. Though the graph convolutional neural network (GCNN) model [41] utilizes both local and nonlocal relations, it is developed for the document-level RE task. More importantly, the intersentence edge in GCNN [41] represents the co-reference or adjacent sentence relation. In contrast, we build the semantic, verb, and entity edges between two sentences, which are easier to get than the co-reference relation and carry more important information than the adjacent sentence relation.

III. Proposed Model

In this section, we first present the problem definition and give the overall structure of the proposed model and then illustrate each component in detail.

A. Problem Definition and Model Overview

Definition 1 (Sentence-Level Relation Extraction): Given a sentence d, a subject entity en_s, and an object entity en_o in d, the task of sentence-level RE is to predict the relation label $y \in \mathcal{R}$ between en_s and en_o, where $\mathcal{R}$ is a set of predefined relations and the "no_relation."

Definition 2 (Semisupervised Relation Extraction): Given a set of labeled and unlabeled relation mentions $L = \{(d_i, en_s^i, en_o^i, y_i)\}_{i=1}^{|D_L|}$ and $U = \{(d_i, en_s^i, en_o^i)\}_{i=1}^{|D_U|}$, respectively, where D_L and D_U, respectively, represents the labeled and unlabeled instances set, and the goal of semisupervised RE task is to learn a mapping function that can fit the labeled training data L and capture the information in the unlabeled data U simultaneously.

Model Overview: In this work, we focus on developing a semisupervised RE method by exploiting the relations between labeled and unlabeled instances. To this end, we propose an LSGI model. The overall structure of LSGI is shown in Fig. 2(a). It consists of four modules.

1) An input module to build the initial graphs and add the newly identified unlabeled samples into the training set at each iteration.
2) A graph update module to refine and update the graphs.
3) A graph interaction module to fuse the lexical and semantic graphs and then to select high-quality samples from unlabeled data.
4) A prediction module to predict labeled and unlabeled samples for model training and data augmentation, respectively.

B. Input Module

The input module prepares data for other components. As we illustrated in Section I, we consider the semantic and lexical relations between labeled and unlabeled

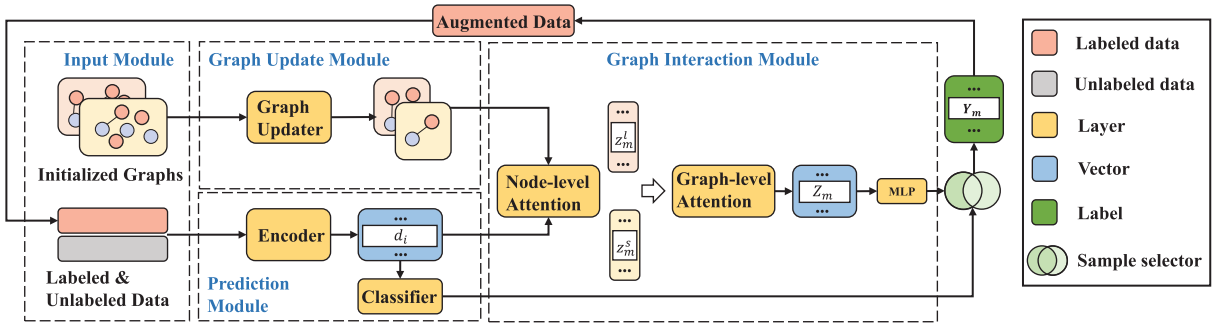


Fig. 2. Architecture of our proposed LSGI model.

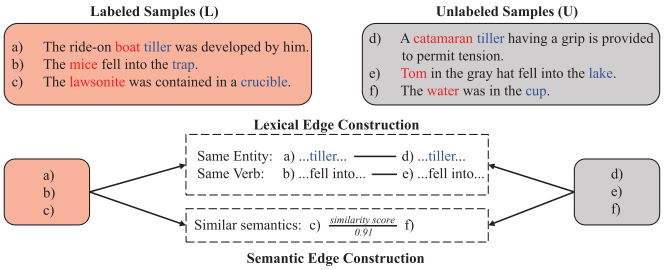


Fig. 3. Construction of the lexical and semantic edges.

instances such that the related instances will be placed closely in the vector space. We first construct two types of graphs, i.e., the lexical reference graph $\mathcal{G}^l$ and the semantic reference graph $\mathcal{G}^s$, where a sample sentence d is now a node in either the labeled set D_L or unlabeled set D_U, each edge represents the lexical or semantic relation between two instances, denoted as $e^l \in E^l$ and $e^s \in E^s$, respectively. More formally, $\mathcal{G}^l = (D_L \bigcup D_U, E^l)$ and $\mathcal{G}^s = (D_L \bigcup D_U, E^s)$.

With the training proceeding, the input module also merges the newly augmented samples into the training set, which is a simple operation. In the following, we present the detail of edge construction.

1) Entity Reference Edge Construction: Entities are critical to RE tasks. To decide whether to add an edge between two sentences by their entity relation, we propose a token-level and type-level similarity. The former denotes there is an overlap between the entity tokens. For example, given a sentence d_1 "The ride-on *boat tiller* was developed by him," there are two entities *"boat"* and *"tiller."* Since two entities in the sentence are adjacent, the other words in the sentence have a weak impact on the entity relation. Now, given another sentence d_2 "A *catamaran tiller* having a grip is provided to permit tension" which also has adjacent entities and an overlapping entity token *"tiller"* with d_1, we will add an edge for d_1 and d_2. The latter denotes that the adjacent entities in two sentences have the same-named entity recognition (NER) typeset recognized by a parser.[1] For example, two sentences "Sentenced to death were *Sulta Hashim Ahmad al-Tai*" and "Voters choosed five republicans to succeed *Rep. Paul Gillmor*" will have an edge since they both have the NER type set {PERSON, TITLE}. Note that only when the entities in one instance are all identified as "Object," will the token-level entity edges be constructed.

Formally, we define the entity relation edge $e_{ij} \in E^{en}$ between two sentences i and j as follows:

$$e_{ij} = \begin{cases} 0, & \mathbb{T}_i \neq \mathbb{T}_j \\ 1, & \mathbb{T}_i = \mathbb{T}_j \quad \text{and } \mathbb{T}_i \neq \varnothing \\ 1, & \mathbb{E}_i \cap \mathbb{E}_j \neq \varnothing \quad \text{and } \mathbb{T}_i = \mathbb{T}_j = \varnothing \end{cases} \tag{1}$$

where $\mathbb{T}$ is the NER typeset, $\mathbb{E}$ is the entity token set for two adjacent entities in the sentence, i is a sample in labeled data, and j is a sample in either labeled or unlabeled data.

2) Verb Reference Edge Construction: When two-entity mentions are not adjacent, the verb or verb phrase (including predicate and nonpredicate) between them plays a critical role in the sentence. For example, in "The *mice* fell into the *trap*," two entities *"mice"* and *"trap"* are not adjacent. In this case, it is the verb phrase *"fell into"* that determines the subject and the object of the action. Actually, the verb and verb phrase play a vital role in a sentence. This is the key property of English and many other languages. In view of this, we construct the verb reference graphs to model the similarity between two samples with the same verb or verb phrase. Specifically, we first select the verb between two entities where the part-of-speech (POS) tag of the verb is in ["VB," "VBN," "VBD," "VBG," "VBP," "VBZ"]. We then judge whether the POS of the verb's next word is in ["IN," "RB," "POS"]. If this is the case, we extract the verb and its next word as the verb phrase. Formally, we define the verb relation edge $e_{ij} \in E^v$ between two sentences i and j as follows:

$$e_{ij} = \begin{cases} 0, & V_i \neq V_j \\ 1, & \text{else} \end{cases} \tag{2}$$

where V denotes the verb or verb phrase in the sentence, i is a sample in labeled data, and j is a sample in either labeled or unlabeled data. Finally, both the entity reference edges and the verb reference edges are presented in the lexical reference graph $\mathcal{G}^l = (D_L \bigcup D_U, E^l)$, where $E^l = E^e \bigcup E^v$ is the edge set consisting of the entity and verb relation edges.

3) Semantic Reference Edge Construction: The semantics denotes the inherent meaning of the sentence beyond the lexical structure. For example, two sentences "The *water* was in a *cup*" and "The *lawsonite* was contained in a *platinum crucible*" are not similar from the lexical point of view. However, their meanings are very similar, and thus, they

[1]https://stanfordnlp.github.io/CoreNLP/index.html

should be placed closely in the vector space. We build a semantic edge $e_{ij} \in E^s$ between two sentences based on their cosine similarity

$$e_{ij} = \begin{cases} 0, & \text{cossim}(d_i, d_j) \leq \delta \\ 1, & \text{else} \end{cases} \tag{3}$$

where d is the sentence embedding, i is a sample in labeled data, j is a sample in either labeled or unlabeled data, and δ is a predefined threshold to choose highly related sentences. The semantic graph $\mathcal{G}^s$ can then be defined as $\mathcal{G}^s = (D_L \bigcup D_U, E^s)$.

To summarize, our reference graph construction is straightforward without complicated rules. The lexical graph is based on the rule of "same entities or same verbs" and the semantic graph is based on the rule of "similar representation of sentences." In this way, we avoid the heavy burden of human intervention. Meanwhile, with the established connections between the labeled and unlabeled data, the lexically or semantically related samples will be positioned closely in the latent vector space, and it becomes easier for the mapping function to classify the unlabeled samples with the labeled samples as their references.

C. Graph Update Module

In semisupervised learning, the training data are gradually augmented with the newly labeled samples. At the initial stage, there are a few training samples and it is hard to learn a good embedding for the sentence. The huge number of unlabeled sentences tends to overwhelm training, which makes the graph model full of noise. To handle the issue, we design a graph update module consisting of an edge importance update component and a semantic edge update component.

1) Edge Importance Update: Since the unlabeled and negative samples (i.e., "no_relation" samples) may contain much irrelevant information, importance sampling can help the model reduce the impact of such information. However, some unlabeled data may be recognized as labeled ones during the training procedure. It is necessary to trace such a difference.

First, we propose a hard importance sampling method to sample important unlabeled sentences. Formally, we define the importance weight $\boldsymbol{W}_{ij}^{\text{un}}$ of the relation edge e_{ij} for unlabeled samples as follows:

$$\boldsymbol{W}_{ij}^{\text{un}} = \begin{cases} 0, & d_i \in D_U \quad \text{and } d_j \in D_U \\ 1, & \text{else.} \end{cases} \tag{4}$$

Second, since the information about negative samples provides no helpful hints for classification, we cut off the connection between negative samples and other samples. Formally, we define the importance weight $\boldsymbol{W}_{ij}^{\text{neg}}$ of the relation edge e_{ij} as for negative samples as follows:

$$\boldsymbol{W}_{ij}^{\text{neg}} = \begin{cases} 0, & d_i \in D_{\text{NEG}} \quad \text{or } d_j \in D_{\text{NEG}} \\ 1, & \text{else} \end{cases} \tag{5}$$

where D_{NEG} represents the negative instances set. The importance sampling strategy not only improves the performance of the proposed model but also reduces the scale of the graph and the computational complexity. With the iteration of the proposed model, useful information is gradually transferred from the training set to the unlabeled dataset. The two types of sampling methods are carried out at the same time; hence, the final importance sampling weight matrix is $\boldsymbol{W} = \boldsymbol{W}^{\text{un}} + \boldsymbol{W}^{\text{neg}}$, which will be used in the graph convolution process.

2) Semantic Edge Update: Since the labeled node-set is gradually enlarged and the representations of the nodes also change with the training, the semantic graph needs to be updated accordingly. We design a two-step strategy to handle this issue.

First, the module will update the graphs by treating the augmented samples as new labeled nodes and adding new edges according to (1)–(3).

Second, the module will remove the edges in the previous semantic graph that are not qualified as a semantic edge in the updated embedding space. The strategy can gradually improve the performance of the semantic reference graph in the later stages of training.

D. Prediction Module

The prediction module is a basic component of semisupervised models that predicts the labeled and unlabeled samples for model training and data augmentation, and it is also used for predicting the test data after the training is finished. In our model, the prediction module consists of two parts: an encoder and a classifier. The encoder aims to encode the samples into latent vectors. There are a number of encoders proposed in the literature, such as LSTM [42], PCNN [14], and PRNN [2]. Since this is not the focus of our research, we simply choose PRNN as the encoder for a fair comparison with the major baseline DualRE [18]. PRNN utilizes various types of text information as features, including POS tags, NER tags, tokens, and entity positions. It then applies a bidirectional LSTM to learn the corresponding representation for each token and uses the attention mechanism to get the final embedding of the sentence.

As for the classifier, we apply the softmax function to the sentence embedding to get the class label of the relation mentions. Note the predictions on labeled data are used for model training with a crossentropy loss function

$$\mathcal{L}_P = -\sum_{l \in L} \boldsymbol{y}_l \cdot \log\left(\tilde{\boldsymbol{y}}_l^P\right) \tag{6}$$

where y_l is the encoded true label, and $\tilde{y}_l^P$ is the logits output by the prediction module P for the l_{th} sample in L. On the other hand, the predictions on unlabeled data will be combined with the output from the graph interaction module to get the final predictions of the unlabeled samples. To clarify, we denote this prediction as $\tilde{\boldsymbol{y}}_u^P = \text{softmax}(\boldsymbol{o}_u^P)$, where $\boldsymbol{o}_u$ is the output for the instance u in the unlabeled data U.

E. Graph Interaction Module

The goal of the graph interaction module is used to deeply fuse the multiple reference graphs. It adopts the hierarchical structure, including a node-level attention layer and a graph-level attention layer.

1) Node-Level Attention Layer: To reduce the human intervention, our reference graphs are constructed roughly based on two simple rules, and the connections may contain many noises. For example, given two sentences "This week, for example, brought a public *statement* by the *head* of NOW," and "Technological advances have brought the *wars* into the *living rooms*," they have the same verb "*brought*" and have an edge between them. However, these two sentences have different relations and the connection information is not consistent with the relation label. In view of this, we propose the node-level attention layer to refine connection information based on the multihead self-attention mechanism [43], [44].

Given a sample node pair in the reference graph $\mathcal{G}$, the node-level attention $\boldsymbol{a}_{ij}^{\mathcal{G}}$ means how important the reference node j is for node i

$$\boldsymbol{a}_{ij}^{\mathcal{G}} = \text{att}_{\text{node}}(\boldsymbol{x}_i, \boldsymbol{x}_j, \mathcal{G}) \tag{7}$$

where att_{node} denotes the node-level attention layer, and $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the feature embeddings from the encoder. Since different reference graphs contain different patterns, att_{node} in different reference graphs share different parameters. Next, we use a softmax function to normalize the obtained attention value. The normalized equation is

$$\boldsymbol{\alpha}_{ij}^{\mathcal{G}} = \text{softmax}\left(\boldsymbol{a}_{i,j}^{\mathcal{G}}\right) = \frac{\exp(\boldsymbol{a}_{ij})}{\sum_{k\in\mathcal{N}_i^{\mathcal{G}}} \exp(\boldsymbol{a}_{ik})} \tag{8}$$

where $\boldsymbol{\alpha}_{ij}^{\mathcal{G}}$ represents the normalized attention coefficient, $\mathcal{N}_i^{\mathcal{G}}$ represents the neighbors of node i in the reference graph $\mathcal{G}$. Once $\boldsymbol{\alpha}_{ij}^{\mathcal{G}}$ is obtained, we aggregate the node i's neighbors in the graph $\mathcal{G}$ with the normalized weights. The graph-based output is then calculated as follows:

$$\boldsymbol{z}_i^{\mathcal{G}} = \sigma\left(\sum_{j\in\mathcal{N}_i^{\mathcal{G}}} \boldsymbol{\alpha}_{ij}^{\mathcal{G}} \boldsymbol{W}\boldsymbol{x}_j\right). \tag{9}$$

Moreover, we utilize the multihead attention to improve stability and K independent attention layer is applied to compute the hidden states. After that, we concatenate the hidden states as follows:

$$\boldsymbol{z}_i^{\mathcal{G}} = \Big\|_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i^{\mathcal{G}}} \boldsymbol{\alpha}_{ij,k}^{\mathcal{G}} \boldsymbol{W}_k \boldsymbol{x}_j\right) \tag{10}$$

where $\boldsymbol{\alpha}_{ij,k}^{\mathcal{G}}$ represents the normalized attention coefficient computed by the kth attention mechanism in the graph $\mathcal{G}$. $\boldsymbol{W}_k$ is the corresponding linear transformation's weight matrix.

Finally, we obtain the corresponding node embedding sets from different reference graphs. Assuming that there are m (=2 in our case) graphs in total, it is expressed as: $\{Z_{\mathcal{G}_1}, Z_{\mathcal{G}_2}, \ldots, Z_{\mathcal{G}_m}\}$.

2) Graph-Level Attention Layer: Different reference graphs contain various types of information and their contributions to the model vary considerably. Therefore, we add a graph-level attention layer as the second layer in the module to adaptively adjust the weights of different graphs. Inspired by the HAN model [45], we propose a method that can automatically learn the importance of different graphs to effectively fuse all reference information. Specifically, the weight of the i_{th} node in the graph $\mathcal{G}$ is calculated as follows:

$$\boldsymbol{b}_i^{\mathcal{G}} = \frac{1}{|\mathcal{N}|} \sum_{i\in\mathcal{N}} \boldsymbol{q} \cdot \text{MLP}\left(\boldsymbol{z}_i^{\mathcal{G}}\right) \tag{11}$$

where $\mathcal{N}$ is the node set and $\boldsymbol{z}_i^{\mathcal{G}}$ is the embedding of the i_{th} node in $\mathcal{G}$, MLP is a multilayer perceptron, and $\boldsymbol{q}$ is the graph-level attention vector randomly initialized and updated in the network.

We then use a softmax function to normalize the weight $\boldsymbol{w}_i^{\mathcal{G}}$

$$\boldsymbol{\beta}_i^{\mathcal{G}} = \text{softmax}\left(\boldsymbol{b}_i^{\mathcal{G}}\right) = \frac{\exp\left(\boldsymbol{b}_i^{\mathcal{G}}\right)}{\sum_{\mathcal{G}=1}^{M} \exp\left(\boldsymbol{b}_i^{\mathcal{G}}\right)} \tag{12}$$

where M is the number of reference graphs. With the normalized graph-level importance weights, we aggregate all the graph-specific embeddings to get the final representation $\boldsymbol{z}_i$ for a node i

$$\boldsymbol{z}_i = \sum_{\mathcal{G}=1}^{M} \boldsymbol{\beta}_i^{\mathcal{G}} \cdot \boldsymbol{z}^{\mathcal{G}_i}. \tag{13}$$

Since each node i has a graph-specific embedding $\boldsymbol{z}_i$, we can feed $\boldsymbol{z}_i$ into a one-layer MLP to get the prediction $\tilde{y}_i$ of the relation mention in i. Once again, for the labeled samples, we utilize cross entropy as the loss function

$$\mathcal{L}_M = -\sum_{l\in L} \boldsymbol{y}_l \cdot \log\left(\tilde{\boldsymbol{y}}_l^M\right) \tag{14}$$

where y_l is the encoded true label, $\tilde{y}_l^M$ is the logits output by the graph interaction module M.

The entire model is trained by alternatively optimizing $\mathcal{L}_P$ and $\mathcal{L}_M$ until the unlabeled data are exhausted. The prediction for an unlabeled sample by this module is denoted as $\tilde{\boldsymbol{y}}_u^M = \text{softmax}(\boldsymbol{o}_u^M)$, which is used to intersect with the outputs generated by the prediction module to get the final label of unlabeled samples.

The overall procedure of the LSGI model is summarized in Algorithm 1.

Algorithm 1 LSGI Algorithm

Input: The labeled set L, the unlabeled set U, the semantic graph $\mathcal{G}^s$, the lexical graph $\mathcal{G}^l$, the k is the number of augmented data in each iteration

Output: The final prediction module P

1: **while** $U \neq \varnothing$ or not converge **do**
2: Train the prediction module p with L and predict the unlabeled samples $Y_U^P \leftarrow U$ using $\mathcal{P}$
3: Update graphs $\mathcal{G}^l, \mathcal{G}^s$ using the graph update module
4: Train the graph interaction module with $\mathcal{G}^l$ and $\mathcal{G}^s$ and predict the unlabeled samples $Y_U^M \leftarrow U$ using $\mathcal{G}$
5: Get the joint prediction $D_l \leftarrow Y_U^M \cap Y_U^P$.
6: Add the top k samples with the highest confidence in D_l to L
7: **end while**
8: Return the final prediction module P.

TABLE I
STATISTICS FOR DATASETS

Dataset	#Train	#Dev	#Test	#Relations	%No_relation
SemEval	7,199	800	2,715	19	17.6
TACRED	68,124	22,631	15,509	42	79.5

IV. EXPERIMENT

A. Dataset

We use two public datasets in our experiments: SemEval and TACRED (Text Analysis Conference (TAC) relation extraction dataset).

1) *SemEval (SemEval-2010 Task8) [1]:* It is the most commonly used dataset for RE tasks. It contains 18 relations like "*Product-Producer (e2,e1)*" and an extra "*no_relation*."
2) *TACRED (TAC Relation Extraction Dataset) [2]:* It is a large-scale RE dataset, which consists of news lines and online texts and is used in the annual TAC KB population competition. It includes 41 relation types, such as "*per:age*" and an extra "*no_relation*."

Note that the relations in SemEval are directional, i.e., $\text{RE}(e_1, e_2) \neq \text{RE}(e_2, e_1)$. The relations in TACRED do not consider direction. The detailed descriptions of datasets are shown in Table I. We present the number of relation mentions in the train/dev/test sets.

We use the data split provided by DualRE [18], i.e., 5%, 10%, and 30% of training data from SemEval, and 3%, 10%, and 15% from TACRED are used as labeled sets, and 50% of training samples are used as unlabeled data for both datasets.

B. Compared Methods and Settings

We compared our LSGI model with the following nine baselines and one method using gold labels.

1) *LSTM [42]:* It is a widely used method in text classification tasks.
2) *AGGCN [13]:* It utilizes GCN with multihead attention to extract important information from the dependency tree. Note that AGGCN does not distinguish the direction of the relationship since it discards the position of entity words.
3) *PCNN [14]:* It employs a piecewise CNN with position embedding to model word sequences.
4) *PRNN [2]:* It utilizes Bi-LSTM to encode position, POS, NER, and token information. It is the state-of-the-art supervised method for RE tasks.
5) *NERO [46]:* It is a rule-based method. It first obtains patterns with manual annotation from the data and then uses soft matching for extracting relations. For a fair comparison, we use both labeled and unlabeled data as a training set for rule matching in NERO.
6) *Self-Training [24]:* It is a traditional semisupervised method. It utilizes the prediction information on unlabeled data and adds the top-ranked samples into the training set.
7) *Self-Training + VAT:* We add the virtual adversarial loss in virtual adversarial training (VAT) [47] upon the base self-training method as a regularization. The virtual adversarial loss is a new measure of local smoothness of the conditional label distribution given input which can smooth the model.
8) *Mean-Teacher [25]:* It is a semisupervised method that utilizes the intersection set of predictions by two different classifiers on unlabeled data and adds the top-ranked samples into the training set. We construct two classifiers using different dropout rates of 0.5 and 0.3, respectively.
9) *Re-Ensemble [48]:* It constructs two independent prediction modules with the same structure and then uses the intersection predicted by these two modules to get high-confidence label information.
10) *DualRE [18]:* It jointly trains a prediction and a retrieval module to select samples from unlabeled data. We use the pointwise variant as it performs better than the other pairwise variant.
11) *LSGI$^-$ (MRefG) [27]:* It is a simplified version of the LSGI model which only has the lexical graph. The model does not take the semantic graph into consideration and does not contain the graph interaction module.
12) *RE-Gold:* It trains PRNN as the base model by using the label information in both labeled and unlabeled data. This represents that labels' overall unlabeled instances can be perfectly predicted, which gives an upper bound performance of semisupervised methods.

Among the baselines, the first four methods and RE-Gold are supervised and the remains are semisupervised ones. For the baselines with released code, we retrain them using the optimal hyperparameters reported in their original papers. For those without source code, we re-implement and retrain them under the same settings as our model. Specifically, we use 300-dimension GloVe [49] as pretrained word embeddings and use the Stanford CoreNLP tool [50] to extract dependency information. Other parameters unique to our model are adjusted by the development set. Specifically, the dimensions of the word embedding, NER, POS, and hidden layer are set to 300, 30, 30, and 200, respectively. The number of heads and the number of dimensions for each head in the graph module is set to 4 and 32, respectively. The learning rate of PRNN is 1.0, and the corresponding learning decay is 0.9. The learning rate of GAT is set to 0.005. The ratio of augmented data generated in each iteration to unlabeled data is 0.1. The threshold δ for constructing a semantic graph is set to 0.9.

Moreover, following DualRE, we choose PRNN as the encoder to encode additional information for all semisupervised methods. The only exception is NERO [46] which uses a Bi-LSTM as encoder since the position and NER information has already been included during preprocessing.

C. Main Results

The comparison results on SemEval and TACRED are shown in Tables II and III, respectively. We divide the results into four parts. The first and the second part are supervised and semisupervised methods, while the third and the fourth parts are our proposed method and the upper bound one. From Tables II and III, we have the following important findings.

1) It is clear that our LSGI model is the best among all methods. It significantly outperforms the best baseline DualRE

TABLE II

COMPARISON RESULTS ON SEMEVAL. THE BEST SCORES ARE IN BOLD, AND SECOND BEST ONES ARE UNDERLINED. ALL RESULTS ARE THE AVERAGE SCORES OF FIVE RUNS WITH RANDOM SEED. "‡" INDICATES THE STATISTICALLY SIGNIFICANT IMPROVEMENTS (I.E., TWO-SIDED T-TEST WITH $p < 0.01$) OVER THE BEST BASELINE

Methods/LabeledData	5%			10%			30%		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
LSTM	24.26 ± 1.76	21.04 ± 1.63	22.52 ± 1.62	40.14 ± 5.08	33.08 ± 4.01	35.88 ± 2.17	64.13 ± 2.46	63.78 ± 3.03	63.84 ± 0.55
AGGCN	8.26 ± 4.44	5.07 ± 0.49	5.99 ± 1.26	9.70 ± 3.51	4.83 ± 0.30	6.34 ± 0.86	8.69 ± 1.93	5.32 ± 0.25	6.60 ± 0.47
PCNN	41.56 ± 2.51	39.30 ± 3.56	40.30 ± 2.49	53.68 ± 1.26	49.87 ± 1.50	51.66 ± 1.38	64.49 ± 0.64	62.81 ± 0.55	63.37 ± 0.42
PRNN	55.65 ± 1.34	53.73 ± 1.25	54.66 ± 0.89	63.47 ± 3.14	61.76 ± 2.20	62.49 ± 0.59	69.66 ± 2.19	68.76 ± 2.60	69.14 ± 1.02
NERO	68.00 ± 1.75	50.63 ± 1.06	58.01 ± 0.19	66.40 ± 0.81	51.96 ± 1.23	58.28 ± 0.52	70.12 ± 0.26	52.28 ± 0.61	59.90 ± 0.27
Self-Training	60.20 ± 1.80	60.03 ± 2.88	60.05 ± 0.75	64.32 ± 1.07	68.97 ± 1.41	66.55 ± 0.83	69.65 ± 0.78	73.68 ± 1.81	71.59 ± 0.79
Self-Training+VAT	59.45 ± 0.71	59.66 ± 1.52	59.54 ± 0.46	65.30 ± 1.09	66.37 ± 0.98	65.82 ± 0.40	70.03 ± 0.51	74.52 ± 1.22	72.20 ± 0.60
Mean-Teacher	63.14 ± 3.13	57.51 ± 1.10	60.15 ± 1.32	66.07 ± 0.93	67.82 ± 0.99	66.93 ± 0.89	69.69 ± 0.84	73.98 ± 1.22	71.76 ± 0.59
RE-Ensemble	60.78 ± 1.12	61.08 ± 1.88	60.90 ± 0.43	65.70 ± 0.55	68.38 ± 0.54	67.02 ± 0.49	71.21 ± 1.25	73.33 ± 1.21	$\underline{72.24 \pm 0.31}$
DualRE	61.11 ± 1.60	63.45 ± 1.29	$\underline{62.24 \pm 0.76}$	65.56 ± 0.35	68.97 ± 1.31	$\underline{67.21 \pm 0.56}$	70.31 ± 1.09	74.04 ± 1.26	72.12 ± 0.51
LSGI$^-$	64.75 ± 1.19	64.99 ± 2.43	64.84 ± 1.11	66.96 ± 0.98	71.37 ± 0.80	69.09 ± 0.44	70.64 ± 1.32	74.90 ± 2.75	72.66 ± 0.69
LSGI	65.16 ± 0.65	65.81 ± 1.61	$\mathbf{65.47^{\ddagger} \pm 0.55}$	67.50 ± 0.62	72.29 ± 0.32	$\mathbf{69.81^{\ddagger} \pm 0.44}$	70.45 ± 1.07	75.66 ± 1.13	$\mathbf{72.95 \pm 0.60}$
RE-Gold(PRNN)	72.25 ± 0.97	75.54 ± 1.98	73.83 ± 0.62	71.40 ± 1.42	76.72 ± 0.64	73.95 ± 0.50	72.98 ± 0.96	78.86 ± 0.76	75.80 ± 0.24

TABLE III

COMPARISON RESULTS ON TACRED. THE BEST AND THE SECOND BEST SCORES ARE IN BOLD AND UNDERLINED. ALL RESULTS ARE THE AVERAGE SCORES OF THREE RUNS WITH RANDOM SEED. "‡" INDICATES THE STATISTICALLY SIGNIFICANT IMPROVEMENTS (I.E., TWO-SIDED T-TEST WITH $p < 0.01$) OVER THE BEST BASELINE

Methods/Labeled Data	3%			10%			15%		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
LSTM	38.76 ± 3.19	18.43 ± 3.78	24.55 ± 2.79	48.72 ± 1.57	44.55 ± 1.63	46.82 ± 0.63	53.83 ± 0.73	46.79 ± 2.11	50.02 ± 0.94
AGGCN	51.95 ± 5.16	38.65 ± 5.51	43.69 ± 1.34	59.80 ± 1.00	51.77 ± 0.95	55.48 ± 0.39	61.30 ± 0.82	52.69 ± 0.30	56.66 ± 0.18
PCNN	53.89 ± 3.29	39.93 ± 2.47	45.39 ± 0.78	64.66 ± 3.16	41.84 ± 2.63	50.42 ± 1.00	66.85 ± 0.43	42.90 ± 1.06	52.25 ± 0.67
PRNN	42.85 ± 1.09	39.52 ± 1.65	41.07 ± 0.51	53.44 ± 2.82	51.77 ± 1.88	52.49 ± 0.64	58.88 ± 2.32	51.30 ± 2.04	54.76 ± 0.93
NERO	47.89 ± 0.58	44.21 ± 0.45	45.97 ± 0.41	47.2 ± 0.93	46.94 ± 0.79	47.07 ± 0.61	51.57 ± 1.40	43.98 ± 1.91	47.48 ± 0.92
Self-Training	53.27 ± 1.08	38.53 ± 1.64	44.70 ± 1.38	57.19 ± 1.10	53.73 ± 1.09	53.39 ± 0.15	59.92 ± 0.64	55.71 ± 1.48	57.72 ± 0.64
Self-Training+VAT	52.96 ± 2.60	43.33 ± 0.35	46.41 ± 0.96	60.95 ± 1.74	52.23 ± 2.75	56.20 ± 1.17	59.99 ± 0.49	56.09 ± 1.02	$\underline{57.97 \pm 0.42}$
Mean-Teacher	58.93 ± 1.37	36.31 ± 0.31	44.93 ± 0.46	61.28 ± 1.01	50.94 ± 0.75	55.62 ± 0.34	61.41 ± 0.38	54.66 ± 1.63	57.82 ± 0.75
RE-Ensemble	54.68 ± 2.99	39.93 ± 1.41	46.07 ± 0.44	59.04 ± 1.27	53.06 ± 0.97	55.87 ± 0.30	60.77 ± 1.11	55.57 ± 0.87	57.81 ± 0.30
DualRE	52.60 ± 0.82	41.64 ± 1.06	$\underline{46.47 \pm 0.27}$	58.78 ± 0.52	54.18 ± 0.33	$\underline{56.38 \pm 0.13}$	59.74 ± 1.24	56.20 ± 0.30	57.92 ± 0.23
LSGI$^-$	51.26 ± 2.07	43.01 ± 1.94	46.71 ± 0.30	59.09 ± 1.88	54.00 ± 1.55	56.40 ± 0.18	59.72 ± 0.62	56.52 ± 0.77	58.07 ± 0.13
LSGI	51.23 ± 2.46	44.99 ± 1.82	$\mathbf{47.85^{\ddagger} \pm 0.12}$	61.63 ± 1.43	53.31 ± 1.21	$\mathbf{57.15^{\ddagger} \pm 0.38}$	59.46 ± 0.86	57.30 ± 0.37	$\mathbf{58.36 \pm 0.25}$
RE-Gold (PRNN)	65.63 ± 1.68	57.47 ± 1.18	61.25 ± 0.04	65.06 ± 4.26	60.18 ± 3.16	62.31 ± 0.27	66.13 ± 1.82	59.62 ± 2.35	62.64 ± 0.49

at $p < 0.01$ on the first two splits. It is better than DualRE on the last split yet the improvement is not significant. This is reasonable because DualRE benefits a lot from a large number of labeled data. In addition, our complete LSGI model also beats the simplified version LSGI$^-$ (MRefG) [27]. On 5% and 10% SemEval and 3% and 10% TACRED, LSGI achieves statistically significant improvements over LSGI with p $<$ 0.05. Moreover, on 3% TACRED, it can also show significant improvements at $p < 0.01$. This proves that the semantic graph and the additional interaction can further enhance the performance. Finally, the gap between LSGI and the upper bound RE-Gold method becomes narrow with the increasing number of labeled data.

2) Among the supervised baselines, PRNN and AGGCN perform the best on SemEval and TACRED, respectively. The extremely poor performance of AGGCN on SemEval can be due to that it is not designed for direction-sensitive RE tasks.

3) The semisupervised methods outperform the supervised ones. DualRE is the best semisupervised baseline owing to the carefully designed retrieval module. However, it is still inferior to our method. This proves that our model benefits from the reference graphs by mapping similar samples to the close position. In contrast to the results in its original paper [46], NERO performs worse than DualRE in our experiment. The reason might be twofold. First, the NERO paper uses all training data, while we use at most (30 + 50)% and (15 + 50)% training data on SemEval and TACRED. Note this is necessary for examining semisupervised learning methods, and it is fair since all methods are under the same setting. Second, the NERO paper adopts Bi-LSTM and attention as the encoder for DualRE without considering POS, NER, and position, and thus, the important information is missing from DualRE.

4) The regularization loss in VAT [47] can improve the performance for all ratios of labeled data on TACRED, but it hurts the performance on 5% and 10% SemEval. These results imply that VAT can enhance the base method when the labeled data are relatively large in a semisupervised learning scenario.

5) When combining the results in Table II and those in Table III, we find that it is more difficult to get improvements with reference information on TACRED. This is because the TACRED dataset is far more complicated than SemEval. It has more types of relations and a more skewed distribution between the positive and negative instances, and the reference information brings both the references and noises simultaneously.

V. ANALYSIS

A. Ablation Study

To investigate the impacts of different reference information in our LSGI model, we conduct the ablation study on both datasets. The results on two datasets are shown in Table IV, where "Original" denotes the results for the original LSGI model, "-Verb," "-Entity," and "-Semantic" denote removing the corresponding information from the original model.

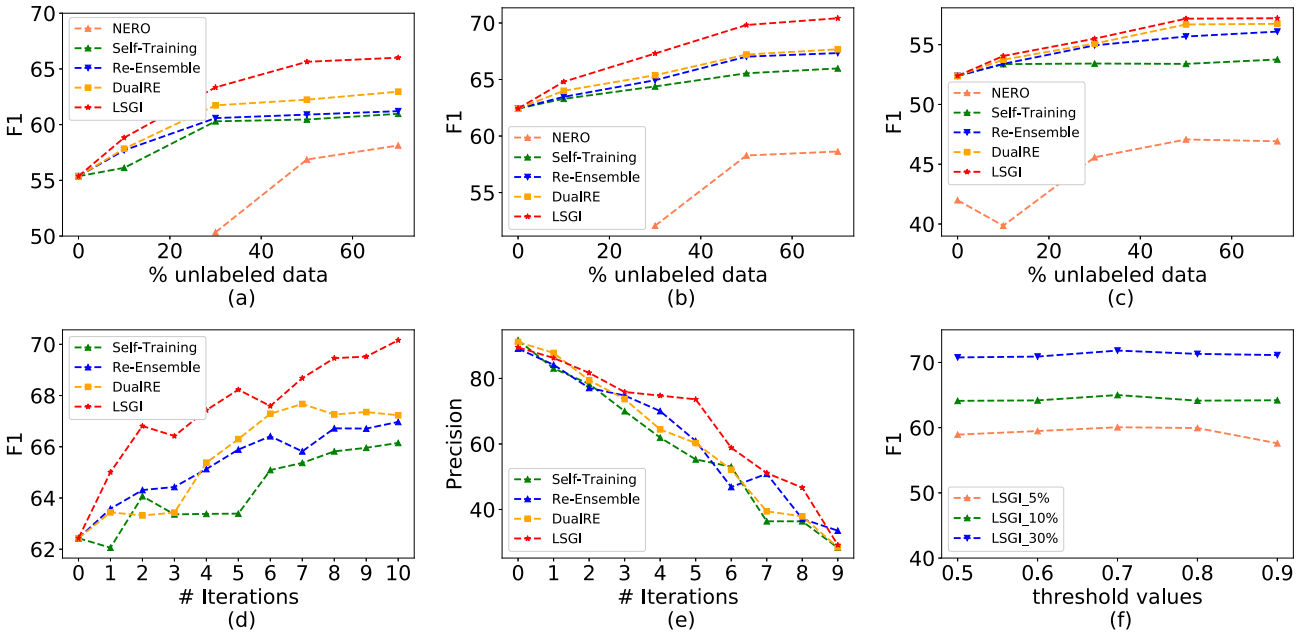


Fig. 4. **Performance on different ratios of unlabeled data:** (a) 5% labeled data on SemEval, (b) 10% labeled data on SemEval, and (c) 10% labeled data on TACRED. **Performance on different iterations of training:** (d) convergence curve of test F1 on SemEval and (e) precision of the augmented samples on SemEval. **Effects of the threshold δ in semantic graph construction:** (f) 5%, 10%, and 30% labeled data on SemEval.

TABLE IV
RESULTS FOR ABLATION STUDY. ↓ DENOTES THE DROP OF F1 SCORE

	10%SemEval			10%TACRED		
	precision	recall	F1	precision	recall	F1
LSGI	67.50	72.29	69.81	61.63	53.31	57.15
-Verb	65.86	68.60	67.08↓	57.23	56.57	56.90↓
-Entity	66.95	70.33	68.59↓	59.09	52.75	55.74↓
-Semantic	66.96	71.37	69.09↓	59.09	54.00	56.40↓

As expected, the performance of all simplified variants drops, showing that each reference information contributes to our model.

B. Parameter Sensitivity Analysis

1) Performance on Different Amounts of Unlabeled Data: We first examine the influence of the size of unlabeled data on semisupervised methods. We vary the unlabeled data size to 0%, 10%, 30%, 50%, and 70% of the training set and keep the labeled data to 5%, 10% for SemEval, and 10% for TACRED datasets. The results are shown in Fig. 4(a)–(c).

As can be seen, our model achieves the best results under various settings. Although the DualRE model performs better than the self-supervised model, the overall effect is still not as good as our model. This further proves the effectiveness of our LSGI model on utilizing the reference information.

Moreover, with the increasing ratio of unlabeled data size, the difference between our model and others becomes clearer. This implies that our model can better leverage the unlabeled data. In addition, note that the F1 score of NERO on SemEval is extremely poor when 0%, and 10% training data are used as the unlabeled set, i.e., its F1 score is 12.41, 12.73, 12.74, and 12.87 for 0%, 10%, 5%, 10% data split, respectively. For clearly presenting the curves for other methods, we omit these two results for NERO.

Finally, we find that the self-supervised model is prone to encounter bottlenecks, i.e., it is hard to get improvements with the increase of unlabeled datasets.

2) Performance on Different Iterations of Training: We then investigate the quality of augmented samples on SemEval. Fig. 4(d) and (e) visualizes the test F1 score and the precision of the augmented samples under a different number of iterations using 10% and 50% labeled and unlabeled data. For clarity, we only show the results on SemEval. In addition, note NERO does not have an iteration process; thus, it is omitted in Fig. 4(d) and (e).

As shown in Fig. 4(d), at the early stage of training, the performance curve of all models has an obvious upward trend, showing that semisupervised learning benefits from the newly added samples. When the training proceeds, the curve of self-training becomes stable, while our LSGI and other two models can still get improvements. In Fig. 4(e), the samples selected by our model are not as good as others at the early stage since the reference graphs contain lots of noises at that time. However, in the middle and late stages of training, the graphs are gradually refined and enlarged. Our model obtains more reference information through reference graphs, which helps our model to select the high-quality samples from the unlabeled data. As a result, the precision of newly selected samples of our model becomes higher than other models.

3) Effects of Threshold in Semantic Graph Construction: Fig. 4(f) shows that too high or too low threshold values will reduce the performance of our model. Furthermore, since the model absorbs the classification information from the graph interaction module during training, the impact of the threshold value is not that significant, i.e., the F1 score of

TABLE V

CASE STUDY. THE RED AND BLUE MARKED TOKENS DENOTE SUBJECT (e_1) AND OBJECT (e_2) ENTITY, AND ✓ AND ✗ DENOTE A CORRECT AND WRONG PREDICTION OF THE RELATION BETWEEN TWO ENTITIES, RESPECTIVELY

Sentence	DualRE	LSGI
S1: Dartmouth nearly narrowed the gap to one goal when a slapshot popped out of the glove of Princeton goalie Zane Kalemba '10.	Entity-Origination (e1,e2)(✓)	Entity-Origination (e1,e2)(✓)
S2: When I got up there, a woman popped out of an office and introduced herself as the PM's secretary.	Member-Collection (e2,e1)(✗)	Entity-Origination (e1,e2)(✓)
S3: However, legal experts expressed surprise at car manufacturer's stance.	No_Relation(✗)	Product-Producer (e1,e2)(✓)
S4: The kind of treatments necessary to kill the bacteria which cause acne breakouts must be prescribed by a doctor.	Cause-Effect (e2, e1) (✗)	Cause-Effect (e1, e2) (✓)

TABLE VI

RESULTS FOR COMPLEXITY ANALYSIS

Methods	10%SemEval			10%TACRED		
	F1	#para	time	F1	#para	time
LSGI	69.81	5.9M	641.42s	57.15	16.4M	2.23h
Self-Training	66.55	5.8M	602.76s	53.39	16.3M	2.01h
Self-Training+VAT	65.82	5.8M	1401.11s	56.20	16.3M	4.95h
Mean-Teacher	66.93	11.6M	1195.16s	55.62	32.6M	4.09h
RE-Ensemble	67.02	11.6M	1103.26s	55.87	32.6M	4.12h
DualRE	67.21	11.6M	1189.79s	56.69	32.6M	4.49h

the model fluctuates in about a 1% span. In most cases, the best performance is achieved when the threshold is around 0.8.

C. Case Study

To have a close look at the impacts of different reference information in our model, we perform a case study by comparing it with the strongest baseline DualRE on four sentences (abbreviated as S1, . . . ,S4). The results are shown in Table V.

S1 is a sample in the training set, and its structure is complex. Both our model and DualRE can correctly classify the relation.

S2 has the same relation class as S1, and interentity verbs provide helpful hints. Nevertheless, DualRE does not obtain helpful information from it and misclassifies S2. In contrast, by building the verb edges between S2 and S1, our model can successfully recognize the importance of "*popped out of*," and obtains the true relation, "*Entity-Origination (e1, e2)*."

In S3, the meaning of the entity pair determines the relation, and DualRE is easily affected by other components in the sentence and cannot identify the important features. Consequently, it makes a wrong prediction for the entity pair ("*car*," "*manufacturer*"). Our model, on the other hand, can assign the correct "*Product-Producer (e1-e2)*" relation to the entity pair. Through deep analysis, we find that S3 has several entity edges connected with the samples in the training set. Here, we give one example: "Try a Google search for the *motherboard manufacturer*." These two sentences both have one adjacent entity and a relation label "*Product-Producer (e1, e2)*." Clearly, it is the entity edge that helps our LSGI model to make a correct prediction.

S4 also has a complex structure. Since most of training samples containing "*cause*" are in the form of "*e1*…caused by…*e2*," their relations are "*Cause-Effect (e2, e1)*," which misleads DualRE about the direction of the relation. In our case, S4 has a semantic edge with a training sample "By avoiding or limiting contact with trash the chance of contracting *germs* that lead to *sickness* or disease is decreased." This sentence has a similar meaning and structure to S4. Moreover, the verb "lead to" in its attributive clause is the synonym of "cause." Both these provide our model with helpful hints to give the correct direction of the relation. When we consider multiple similar samples together, we can reduce the probability of misclassification. These results are in line with our expectations.

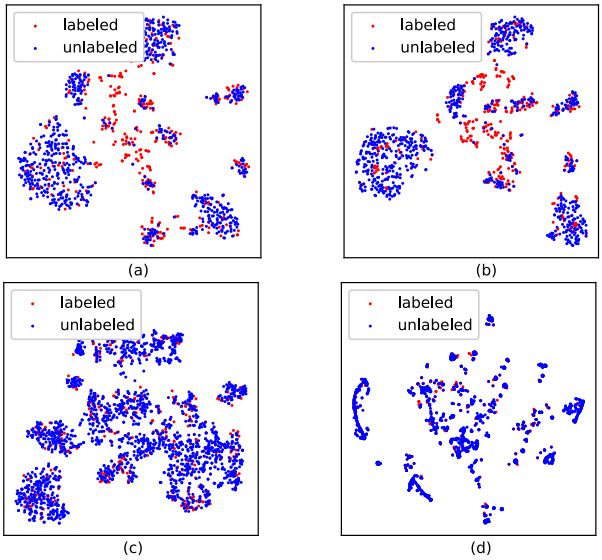


Fig. 5. **Visualization on SemEval and TACRED.** Each point indicates one sample. Closer the labeled and unlabeled data points denote better performance. (a) Visualization of DualRE with 5% labeled data on SemEval. (b) Visualization of LSGI with 5% labeled data on SemEval. (c) Visualization of DualRE with 3% labeled data on TACRED. (d) Visualization of LSGI with 3% labeled data on TACRED.

D. Computational Complexity Analysis

In order to gain more insights into the computational complexity and the tendency of running time, we compare our method with baselines in terms of performance, parameters,

and running time in Table VI. Our platform is a 24-GB NVIDIA RTX 3090 GPU.

It is clear that our LSGI has fewer parameters than almost all baselines except self-training which contains only one prediction module, while other methods need two modules. Please note that all methods adopt PRNN as the encoder, and the parameters in the embedding layer of PRNN constitute the main part of the parameter space. While the baselines with two prediction modules employ two separate PRNN encoders, LSGI shares one PRNN encoder between these two modules, and thus, it has the least parameters. In summary, our model has achieved performance improvements without increasing the number of parameters and running time.

E. Visualization

To examine the impacts of the learned feature representations for LSGI and DualRE qualitatively, we provide samples selected after the first iteration of the corresponding two models and visualize them by t-SNE [51], which can visualize high-dimensional data by giving each data point a position in a 2-D or 3-D map, and produces significantly better visualizations by reducing the tendency to crowd points together in the center of the map. As a result, each sample is mapped as a vector in a 2-D space. Note that due to a large number of negative samples in the TACRED, we have removed the negative samples.

One can clearly observe that the advantage of our proposed model over DualRE. As shown in Fig. 5, the vector obtained by DualRE is relatively loose in the vector space, and the connection between labeled and unlabeled samples is not close enough. In contrast, our model successfully links the labeled samples with unlabeled ones and these two types of samples are located nearby in the vector space.

VI. Conclusion

In this article, we propose an LSGI model for semisupervised RE. The main idea is to provide the reference information for the unlabeled samples before they are sent into the classifier. To this end, we construct the lexical and semantical graphs to connect labeled and unlabeled samples based on two simple rules such that similar instances are positioned closely in the encoded vector space. Moreover, these two types of graphs are combined together to get a graph-specific representation, which will be used to assist the prediction task. As a result, the mapping function can better predict the relation label for the referees. By choosing the high-confidence samples from the unlabeled corpus, the training set is successfully augmented during each iteration of the semisupervised learning process. The extensive experimental results on two public datasets demonstrate that our proposed model consistently outperforms the state-of-the-art baselines.

Acknowledgment

The numerical calculations in this article have been done on the supercomputing system at the Supercomputing Center, Wuhan University, Wuhan, China.

References

[1] I. Hendrickx *et al.*, "SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," in *Proc. Workshop Semantic Evaluations: Recent Achievements Future Directions (DEW)*, 2009, pp. 33–38.

[2] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, "Position-aware attention and supervised data improve slot filling," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 35–45.

[3] Y. Yao *et al.*, "Docred: A large-scale document-level relation extraction dataset," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 764–777.

[4] N. Peng, H. Poon, C. Quirk, K. Toutanova, and W. Yih, "Cross-sentence n-ary relation extraction with graph LSTMs," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 101–115, Dec. 2017.

[5] L. B. Soares, N. FitzGerald, J. Ling, and T. Kwiatkowski, "Matching the blanks: Distributional similarity for relation learning," in *Proc. ACL*, 2019, pp. 2895–2905.

[6] Z. Zhang *et al.*, "Distilling knowledge from well-informed soft labels for neural relation extraction," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2020, vol. 34, no. 5, pp. 9620–9627.

[7] C. De Sa *et al.*, "Incremental knowledge base construction using DeepDive," *VLDB J.*, vol. 26, no. 1, pp. 81–105, Feb. 2017.

[8] C. Quirk and H. Poon, "Distant supervision for relation extraction beyond the sentence boundary," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2017, pp. 1171–1182.

[9] M. Yu, W. Yin, K. S. Hasan, C. dos Santos, B. Xiang, and B. Zhou, "Improved neural relation detection for knowledge base question answering," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 571–581.

[10] A. Kadry and L. Dietz, "Open relation extraction for support passage retrieval: Merit and open issues," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 1149–1152.

[11] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," *J. Mach. Learn. Res.*, vol. 3, pp. 1083–1106, Feb. 2003.

[12] W. Zeng, Y. Lin, Z. Liu, and M. Sun, "Incorporating relation paths in neural relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 1768–1777.

[13] Z. Guo, Y. Zhang, and W. Lu, "Attention guided graph convolutional networks for relation extraction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 241–251.

[14] D. Zeng, K. Liu, Y. Chen, and J. Zhao, "Distant supervision for relation extraction via piecewise convolutional neural networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1753–1762.

[15] M. Reiplinger, M. Wiegand, and D. Klakow, "Relation extraction for the food domain without labeled training data—Is distant supervision the best solution?" in *Advances in Natural Language Processing* (Lecture Notes in Computer Science), vol. 8686. Cham, Switzerland: Springer, 2014, pp. 345–357.

[16] G. Ji, K. Liu, S. He, and J. Zhao, "Distant supervision for relation extraction with sentence-level attention and entity descriptions," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 3060–3066.

[17] W. Jia, D. Dai, X. Xiao, and H. Wu, "ARNOR: Attention regularization based noise reduction for distant supervision relation classification," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1399–1408.

[18] H. Lin, J. Yan, M. Qu, and X. Ren, "Learning dual retrieval module for semi-supervised relation extraction," in *Proc. World Wide Web Conf.*, 2019, pp. 1073–1083.

[19] R. Jones, A. McCallum, K. Nigam, and E. Riloff, "Bootstrapping for text learning tasks," in *Proc. Workshop Text Mining, Found., Techn. Appl. (IJCAI)*, 1999, pp. 52–63.

[20] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *Proc. 5th ACM Conf. Digit. Libraries (DL)*, 2000, pp. 85–94.

[21] D. S. Batista, B. Martins, and M. J. Silva, "Semi-supervised bootstrapping of relationship extractors with distributional semantics," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 499–504.

[22] P. Gupta, B. Roth, and H. Schütze, "Joint bootstrapping machines for high confidence relation extraction," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, M. A. Walker, H. Ji, and A. Stent, Eds., 2018, pp. 26–36.

[23] Z. Wang *et al.*, "Learning from explanations with neural execution tree," in *Proc. ICLR*, 2020, pp. 1–18.

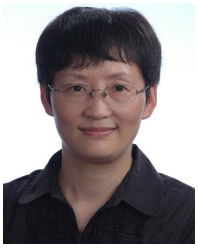
[24] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," in *Proc. 7th IEEE Workshops Appl. Comput. Vis. (WACV/MOTION)*, Jan. 2005, pp. 29–36.

[25] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. ICLR*, 2017, pp. 1–16.
[26] J. R. Curran, T. Murphy, and B. Scholz, "Minimising semantic drift with mutual exclusion bootstrappin," in *Proc. PACLING*, 2007, pp. 172–180.
[27] W. Li, T. Qian, X. Chen, K. Tang, S. Zhan, and T. Zhan, "Exploit a multi-head reference graph for semi-supervised relation extraction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–7.
[28] D. M. Bikel, R. M. Schwartz, and R. M. Weischedel, "An algorithm that learns what's in a name," *Mach. Learn.*, vol. 34, nos. 1–3, pp. 211–231, 1999.
[29] A. McCallum, D. Freitag, and F. C. N. Pereira, "Maximum entropy Markov models for information extraction and segmentation," in *Proc. ICML*, 2000, pp. 591–598.
[30] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001, pp. 282–289.
[31] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proc. EMNLP*, Jul. 2012, pp. 1201–1211.
[32] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proc. 1st Workshop Vector Space Model. Natural Lang. Process.*, 2015, pp. 39–48.
[33] P. Zhou *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 207–212.
[34] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017, pp. 1–14.
[35] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, pp. 1024–1034.
[36] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," 2018, *arXiv:1812.08434*.
[37] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Dec. 2009.
[38] X. Wang *et al.*, "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, May 2019, pp. 2022–2032.
[39] T.-J. Fu, P.-H. Li, and W.-Y. Ma, "GraphRel: Modeling text as relational graphs for joint entity and relation extraction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1409–1418.
[40] F. Christopoulou, M. Miwa, and S. Ananiadou, "A walk-based model on entity graphs for relation extraction," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 81–88.
[41] S. K. Sahu, F. Christopoulou, M. Miwa, and S. Ananiadou, "Inter-sentence relation extraction with document-level graph convolutional neural network," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4309–4316.
[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[43] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[44] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2018, pp. 1–12.
[45] X. Wang *et al.*, "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, May 2019, pp. 2022–2032.
[46] W. Zhou *et al.*, "NERO: A neural rule grounding framework for label-efficient relation extraction," in *Proc. WWW*, 2020, pp. 2166–2176.
[47] T. Miyato, S. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1979–1993, Jul. 2019.
[48] G. French, M. Mackiewicz, and M. H. Fisher, "Self-ensembling for visual domain adaptation," in *Proc. ICLR*, 2018, pp. 1–20.
[49] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.
[50] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The Stanford corenlp natural language processing toolkit," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 55–60.
[51] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, Nov. 2008.

Wanli Li (Member, IEEE) received the B.S. degree in information management and information system from Guangdong Medical University, Zhanjiang, China, in 2016, and the M.S. degree in circuits and systems from South China Normal University, Guangzhou, China, in 2019. He is currently pursuing the Ph.D. degree with the School of Computer Science, Wuhan University, Wuhan, China.

His research interests include natural language processing, relation extraction, and semisupervised learning.

Mr. Li is a member of the ACM.



Tieyun Qian (Member, IEEE) received the B.S. degree in computer science from Wuhan University of Technology, Wuhan, China, in 1991, and the Ph.D. degree in computer science from the Huazhong University of Science and Technology, Wuhan, in 2006.

She is currently a Professor with the School of Computer Science, Wuhan University, Wuhan. She has authored or coauthored over 60 articles in leading conferences and top journals, including the Annual Meeting of Association for Computational Linguistics (ACL), the Conference on Artificial Intelligence (AAAI), the International Conference on Information and Knowledge Management (CIKM), *ACM Transactions on Information Systems* (TOIS), and *ACM Transactions on Knowledge Discovery from Data* (TKDD). Her current research interests include text mining, Web mining, and natural language processing.

Dr. Qian is a member of the ACM and the China Computer Federation (CCF). She has served as a Program Committee Member of many premium conferences, such as the World Wide Web Conference (WWW), AAAI, the International Joint Conference on Artificial Intelligence (IJCAI), and the International Conference on Database Systems for Advanced Applications (DASFAA).

Ming Zhong received the Ph.D. degree in computer science from Wuhan University, Wuhan, China, in 2009.

Since 2010, he has been with Wuhan University as an Associate Professor with the School of Computer Science. His current research interests include big data management and analytics, graph database theory and applications, graph data-oriented machine learning theory and applications, and social network analytics.

Xu Chen was born in 1982. He received the bachelor's degree from the School of Computer Science, Wuhan University, Wuhan, China, in 2004, and the Ph.D. degree from the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, in 2010.

He is currently an Associate Professor with the School of Computer Science, Wuhan University. His research interests include spatial information processing and information retrieval.