Exploit Feature and Relation Hierarchy for Relation Extraction

Mi Zhang, *Student Member, IEEE*, Tieyun Qian 🆔 , *Member, IEEE*, and Bing Liu 🆔 , *Fellow, IEEE*

Abstract—Existing methods in relation extraction have leveraged the lexical features in the word sequence and the syntactic features in the parse tree. Though effective, the lexical features extracted from the successive word sequence may introduce some noise that has little or no meaningful content. Meanwhile, the syntactic features are usually encoded via graph convolutional networks which have restricted receptive field. In addition, the relation between lexical and syntactic features in the representation space has been largely neglected. To address the above limitations, we propose a multi-scale representation and metric learning framework to exploit the feature and relation hierarchy for RE tasks. Methodologically, we *build a lexical and syntactic feature and relation hierarchy* in text data. Technically, we first develop *a multi-scale convolutional neural network* to aggregate the non-successive lexical patterns in the word sequence. We also design *a multi-scale graph convolutional network* to increase the receptive field via the coarsened syntactic graph. Moreover, we present *a multi-scale metric learning* paradigm to exploit both the feature-level relation between lexical and syntactic features and the sample-level relation between instances with the same or different classes. Extensive experiments on three public datasets for two RE tasks prove that our model achieves a new state-of-the-art performance.

Index Terms—Multi-scale feature learning, multi-scale metric learning, relation extraction.

I. INTRODUCTION

THE goal of relation extraction (RE) is to predict the semantic relations between/among entities in the text. RE plays a vital role in many applications such as question answering [1]–[3], knowledge graph [4]–[6], and recommender systems [7], [8]. As a result, it has aroused tremendous research interests in recent years.

There are mainly two types of RE tasks, namely, the binary relation extraction in single sentences [9]–[11] and the cross-sentence n-ary relation extraction [12]–[14]. Fig. 1 shows examples of a binary relation and a ternary relation. For sentence-level RE, given two entities 'He' and 'table,' we aim to extract the
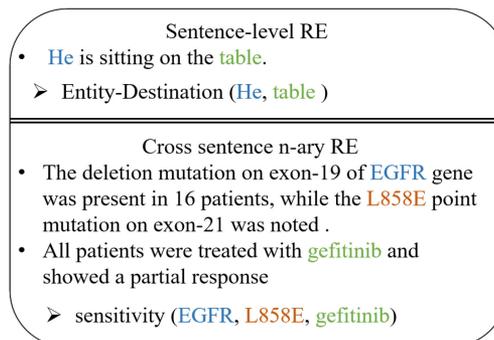
Fig. 1. Examples of a binary relation in one sentence and a ternary relation across two sentences.

gold relation 'Entity-Destination'. For cross-sentence n-ary RE, given three target entities 'EGFR,' 'L858E,' and 'gefitinib' in a piece of text consisting of two sentences, we should extract the gold ternary relation 'sensitivity'.

Current studies in RE focus on feature learning that discovers the representations from raw data. The word sequence has abundant lexical information and the dependency graph of the sentence carries unique syntactic information and they can both make contributions to RE tasks. Consequently, existing RE models can be roughly categorized into two classes: sequence-based and dependency-based. Sequence-based models [15]–[19] extract local and global lexical features from the word sequence mainly using convolutional neural network (CNN) [20] and long short-term memory (LSTM) [21], both of which are crucial for RE tasks. Parse trees can shorten the long-range syntactic relations, and dependency based models apply tree- (graph-) LSTM or graph convolution network (GCN) to parse trees to obtain the syntactic representations of all words [22]–[24], where GCNs are widely adopted due to their superior performance. There are also a few methods [24]–[27] combine both lexical and syntactic features. All the above methods employ the static word embeddings as the input. Recently, several studies [28]–[31] employ the pre-trained language models (PLMs) like BERT [32] to get the contextualized word embeddings which can further improve the performance.

Despite of the effectiveness, existing methods face several weaknesses.

- CNNs based methods [16], [17] extract successive local features in a word by word manner like 'He-is' and 'is-sitting'. This, however, may introduce noise since languages often contain some tokens that have little or no
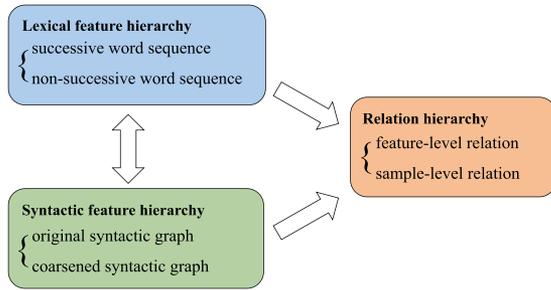
Fig. 2. Feature and relation hierarchy in the text.

meaningful content. For example, given the sentence 'He is sitting on the table,' the auxiliary verb 'is' and the determiner 'the' do not have concrete meanings.

- GCNs [24], [25] have restricted receptive field since they convolute along the dependency path. In the aforementioned example, the goal is to extract the 'Entity-Destination' relation 'on' between two entities 'He' and 'table'. However, 'He' cannot be aggregated with 'on' via a standard convolution operation since they are located in different paths, and simply adding convolutional layers means performing more Laplacian smoothing [33] and will make the output features similar.

- None of existing methods has ever taken the geometric relation between the lexical and syntactic features into account. Indeed, one sentence's lexical and syntactic representations refer to the same word sequence. Their relative position in the feature space should be close and such a property should have a good fit in the encoded representation vectors.

To tackle the above limitations, we propose to construct a feature and relation hierarchy in the text data. As shown in Fig. 2, the *lexical feature hierarchy* consists of the successive and non-successive word sequences, while the *syntactic feature hierarchy* contains the original and coarsened syntactic graphs, and the *relation hierarchy* includes the feature-level and sample-level relations.

Based on the constructed hierarchy, we further present *a multi-scale feature and metric learning framework* to fully exploit these three types of hierarchy for RE tasks. We first design *a multi-scale convolutional neural network* (*MSCNN*) for extracting non-successive local patterns like 'He-sitting' in the word sequence .[1] MSCNN contains a horizontal dilated convolution layer and a vertical convolution layer. The dilated horizontal convolution enables the model focus on the important yet non-successive words in the sentence. The vertical convolution captures the dimension-level sequential patterns which can aggregate every dimension in other words' latent representations.

We then develop *a multi-scale graph convolution network* (*MSGCN*) to obtain the coarsened and refined view of the parse tree. Instead of merely aggregating one- or two-hop neighborhood syntactic information, MSGCN enlarges the receptive field for the word, and thus more local syntactic information can

---

[1]LSTM may focus on 'He' and 'sitting' via attention mechanism. However, LSTM is often used for extracting global information.

be closely related. For example, 'He' and 'table' are merged into a hypernode in the coarsened graph where the 'He' node can see the 'table' node's neighboring words. As a result, the neighbors of 'table' like 'on' can be aggregated with 'He' in one convolution operation.

We also present *a multi-scale metric learning paradigm* (*MSML*) to explore both the feature-level and sample-level relations, such that we can pull the neighbors in the feature and relation space close and push those non-neighbors apart. On one hand, both the syntactic and lexical features of a sentence refer to the same word sequence. Though their representations might be different, their distance in the feature space should be smaller than that of this lexical embedding to the syntactic embedding of another sentence. On the other hand, for the RE task, the samples with same/different relations should be located nearby/apart from each other in the relation space.

Our model is among the first to exploit the hierarchy nature in lexical, syntactic, and relational representations of the text data for RE tasks. To validate our proposed model, we conduct extensive experiments on three popular datasets for the sentence-level and cross-sentence n-ary RE tasks. With either the static word embedding GloVe [34] or the contextualized word embedding BERT and RoBERTa [32], [35], our model outperforms the state-of-the-art counterpart methods for both tasks on all three datasets.

II. RELATED WORK

We briefly review the literature in relation extraction, multi-scale representation learning, and metric learning.

A. Relation Extraction Models

According to the features used in the models, most existing relation extraction methods can be categorized into lexicon based (aka sequence based) and syntax based (aka dependency based) ones.

Early statistical RE models use the relational structure of the verbs to discriminate relations [36], [37], or employ similarity of context word as the features or kernels [10], [38], [39]. Recently, several types of neural models, e.g., CNN based [15], [16], [40], [41], RNN based [42], [43], and LSTM based [19], [44], have been proposed to extract better lexical features from word sequences.

The syntactic information is also effective in improving relation extraction performance as it can capture long-distance relations. Various models have been developed to incorporate syntactic information. For example, LSTM and entity position-aware attention [4] using POS tags, tree or graph structure are employed as part of word embeddings. Tree or graph structures [18], [24], [25], [27], [45] are used to encode dependency relations as embeddings where GCN becomes the mainstream due to its superior performance. Several methods [4], [18], [23], [46] pay attention to the pruning strategies for selecting relevant partial dependency structures. More recently, dependency forest [47] or multiple dependency sub-graphs [26] are constructed to obtain more informative features.

Besides the above sequence and dependency based methods which adopt the static word embeddings as input, several recent studies employ the contextualized word embedding like BERT or RoBERTa for RE tasks based on the fine-tuning or post-training techniques. For example, LUKE [30] uses an improved transformer architecture with an entity-aware self-attention mechanism to enhance entity-related representations. R-BERT [29] incorporates information from the target entities to tackle the relation classification task. CP [31] presents the entity-masked contrastive post-training framework for RE using external knowledge base. SpanRel [48] proposes a single modeling framework that first extracts spans and predicts their labels, then predicts relations between spans.

Overall, existing methods either capture information word by word sequentially or do not fully exploit the dependency along different syntactic paths. The sequence based methods extract successive information from the word sequence and thus may introduce some noise. Meanwhile, the dependency based GCN methods can only convolute along the syntactic paths. In contrast, we present the multi-scale CNN and multi-scale GCN which capture the non-successive information in the sentence and expand the receptive field for the nodes in the parse tree. Also note that the utilization of contextualized word embedding is not the focus of our paper, hence we design a simple post-training method and integrate its output into the input embedding layer of our model.

B. Multi-Scale Representation Learning

Multi-scale information is of great significance in computer vision area. For example, FPN [49] sends images of different scales to the network to extract the features of different scales for integration. D-CNN [50] develops a convolutional network module that aggregates multi-scale contextual information without losing resolution or analyzing re-scaled images.

Some work has also been proposed for learning multi-scale information from graphs. For example, Chen *et al.* [51] propose a coarsening procedure to construct a coarsened graph of smaller size and use graph network to encode graph node embedding, and they then use a refining procedure to get the original graph embedding. HGCN [52] repeatedly aggregates structurally similar nodes into hypernodes and then refines the coarsened graph to the original one.

Inspired by these studies, we introduce the multi-scale learning technique into the natural language processing field. To this end, we propose to build a feature and relation hierarchy in text and develop a novel multi-scale feature and metric learning paradigm tailored for RE tasks.

C. Metric Learning

Metric learning [53] produces distance metrics that capture important relationships among data and is an indispensable technique for many machine learning applications, including image classification, protein function prediction, and recommender systems [54]–[58].

The object of metric learning is to reduce the distance between similar pairs and increase the distance between dissimilar pairs. The triplet loss based optimization essentially learns a distance metric that pulls all similar pairs together, and pushes dissimilar pairs apart. Metric learning is usually applied to face recognition [59] to determine which image belongs to the same person. It is also adopted in recommendation algorithms to compare which product is more suitable for the user [60].

We apply metric learning to RE tasks, and extend it into a multi-scale version that includes the feature-level and sample-level metric learning. To our knowledge, there is no exploration towards this direction in RE. Though CL [31] adopts a contrastive objective in learning, it works on the relations. In contrast, our model learns the relation between the syntactic and lexical representations.

III. PRELIMINARY

A. Task Definition

Relation extraction tasks are mainly categorized into two types: sentence-level and cross-sentence n-ary RE.[2] Formally, let $T = [t_1,..., t_m]$ be a text consisting of sentence(s) with m tokens (words), and $e_1,..., e_n$ be the entity mentions in T. $R = [r_1,..., r_{o-1}, r_o]$ $(r_o = \text{None})$ is a predefined relation set. The relation extraction task can be formulated as a classification problem of determining whether a relation $r \in R$ holds for $e_1,..., e_n$. More specifically, sentence-level RE focuses on the binary relations where two entities occur in the same sentence (i.e., $n = 2$ and T is a sentence) while cross-sentence n-ary RE focuses on multiple sentences with entities (i.e., $n > 2$ and T contains multiple sentences).

B. Anchor Triplet

We sample a positive and negative sample for an anchor to form an anchor triplet (S, S^+, S^-) used in metric learning:

S: the anchor sample, which is currently going to be classified;

S^+: the positive sample having the same class label with S;

S^-: the negative sample having the different class label with S.

Each sample in the anchor triplet contains a word sequence and a dependency graph of an instance in training set. The anchor triplet will be used for metric learning. Our goal of metric learning is that the trained model can reduce the distance between samples from the same class (S, S^+) and increase the distance between the different classes (S, S^-).

C. Preparing Input Representation

For a thorough comparison with various methods, we have prepared two types of word embeddings as the input of our model. One is the static GloVe embedding [34] that is

---

[2]There are several studies towards document-level RE where the relations are expressed across multiple paragraphs, and extra efforts are required to deal with the multiple mentions of an entity. Its learning focus is different from that of sentence-level and cross-sentence n-ary RE and thus we omit it.

widely used in sequence based and dependency based methods. The other is the contextualized embedding including BERT [32] and RoBERTa [35] used in pre-training language model (PLM) based methods where the word embeddings are further refined via fine-tuning or post-training in the downstream tasks.

1) Static Word Embedding: Let $\mathbf{E}_w \in \mathbb{R}^{|V| \times da}$ be the static word embedding [34] lookup table, where $|V|$ is the vocabulary size and da is the dimension of word embedding. $\mathbf{E}_w$ is used to map the text T to a list of word vectors $[\boldsymbol{t}_1^w,..., \boldsymbol{t}_m^w] \in \mathbb{R}^{m \times da}$. Let $\mathbf{E}_o \in \mathbb{R}^{m \times db}$ and $\mathbf{E}_{pos} \in \mathbb{R}^{m \times dc}$ be the position and POS embedding lookup table with random initialization, where db and dc are the dimension of position and POS embedding, respectively. $\mathbf{E}_o$ and $\mathbf{E}_{pos}$ are used to map the position of tokens and the POS tags of tokens to embeddings.

Overall, each token t_i in T is vectorized as $\boldsymbol{t}_i^{wop} = \boldsymbol{t}_i^w \oplus \boldsymbol{t}_i^o \oplus \boldsymbol{t}_i^{pos} \in \mathbb{R}^{da+n \times db+dc}$ containing word, position, and POS information. We use $\boldsymbol{t}^{wop} = [\boldsymbol{t}_1^{wop}, ..., \boldsymbol{t}_m^{wop}]$ as the input for the token in word sequence and dependency graph.

2) Contextualized Word Embedding: Generally, the methods with contextualized embeddings achieve significantly better performance over those with static embeddings. However, this can be largely due to the power of PLMs rather than the methods themselves. In view of this, we design a simple post-training task on RE training set to obtain contextualized word embeddings for RE tasks. Our method is adapted from previous models with slight modification. This ensures that we can pay more attention to the design of multi-scale feature and metric learning. Below we present the detail.

Firstly, to make PLMs capture the location information of entities, we follow the method in [28], [29] by inserting a special token '[E1]' and '[E2]' at both the beginning and the end of the first and the second entity, respectively. We also add '[CLS]' at the beginning of sentences. For example, after inserting the special separate tokens, the running sample sentence with target entities 'He' and 'table' is transformed into:

[CLS] [E1] He [E1] is sitting on the [E2] table [E2].

For each instance in the dataset, we feed the transformed token sequence into PLMs including BERT [32] and RoBERTa [35] and obtain the hidden state output of each token.

Secondly, we perform the post-training task on RE training set, which consists of an entity type prediction task and a relation classification task. Each entity has its own entity type information. For example, in the PubMed dataset, there are three entity types: drug, gene, and mutation. We predict the entity type for each entity with its own embedding and the special token embedding '[E1]' and '[E2]'. Specifically, we apply the average operation to get the entity vector representation for each of the entities, and we then use this entity vector to predict its entity type. For the post-training relation classification task, we concatenate each entity's hidden embedding and the final hidden state vector of the first token '[CLS],' and then add a fully connected layer and a softmax layer for classification. Besides, the predicted entity type embeddings are also added onto the entity embeddings to provide more information for the entities.

After post-training with these two tasks, the output token embeddings will serve as the inputs for the tokens in word sequence and dependency graph.

IV. PROPOSED MODEL

A. Model Overview

In this section, we give an overview of our model. We show its architecture in Fig. 3. As can be seen, our model is composed of three main modules: *MSFL*, *MSML*, and *MT*.

MSFL is a multi-scale feature learning module which contains two components, denoted as C_1 and C_2 in two dotted rectangles. It takes a set of triplets $[S, S^+, S^-]$ as the input and output three types of representations that used for multi-scale metric learning (*MSML*) and the main classification task (*MT*). C_1 adopts a normal Bi-LSTM to get global feature of the sample. C_2 applies the proposed MSCNN and MSGCN to sequence and dependency graph to capture multi-scale local features.

MSML is used for multi-scale metric learning, including the feature level ML that enhances/lessens the relation between the lexical and syntactic features of the same/different sentence, and the relation level ML which strengthens/weakens the relation representation of the same/different classes.

MT performs the main relation classification task based on the combined joint representation from *MSFL*.

B. Multi-Scale Feature Learning (MSFL) Module

The word sequence has abundant lexical information and the dependency graph carries unique syntactic information, both of which can make contributions to RE tasks. CNNs and GCNs have shown effectiveness in extracting lexical and dependency information. However, as we analyze them in the introduction part, CNNs tend to extract successive local features and GCNs have limited receptive fields. In order to tackle these problems, we propose in this section our MSFL module, which consists of the multi-scale convolution neural network (MSCNN) and multi-scale graph convolution network (MSGCN).

1) Multi-Scale Lexical Feature Learning: The core idea in CNNs is to capture local features. A convolution operation involves a filter which is applied to a sliding window composed of several words to produce a new feature. As shown in Fig. 4(a), for an ordinary CNN, when the filter size is 2, the window slides over the sequence and convolutes the successive 2-gram such as 'He-is' in blue rectangle and 'is-sitting' in green rectangle at each step. However, the word 'is' is an auxiliary verb which does not have concrete meaning. Indeed, it is better to put 'He' and 'sitting' in one window given the fixed filter size 2. To this end, we apply the dilated convolution method D-CNN [50] to our RE task, and extend it by convoluting along *both the vertical and horizontal directions* to systematically aggregate lexical information. We term this two-dimensional convolution as MSCNN.

The two-dimensional convolution on sentences is performed with three different convolution kernels (kernel size = [3,4,5])
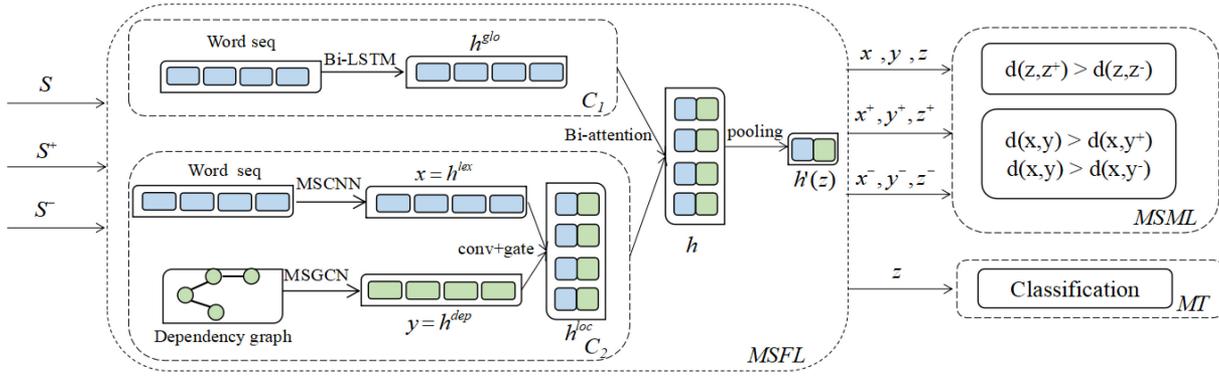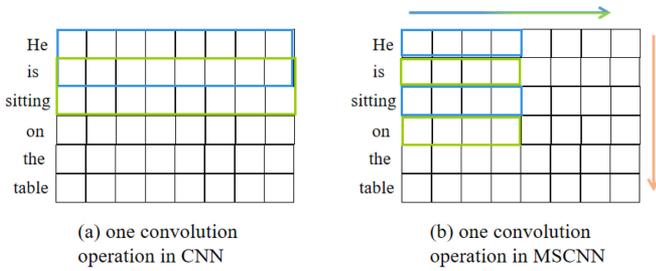
Fig. 3. Architecture of our model.

Fig. 4. Sample of original CNN and our proposed MSCNN.

(a) one convolution operation in CNN

(b) one convolution operation in MSCNN

(a) original syntactic graph

(b) coarsened syntactic graph

Fig. 5. Sample of the original syntactic graph in GCN and the coarsened syntactic graph in our MSGCN.

on word-dimension (horizontal direction) and one convolution kernel (kernel size = d/2, where d is the dimension of input representation) on vector-dimension (vertical direction). On one hand, the horizontal dilated convolution enables the non-successive tokens like 'He' and 'sitting' to be convoluted in one sliding window. On the other hand, the vertical convolution captures the dimension-level sequential patterns such that the corresponding dimensions of the words in the same window can be aggregated. The illustration of MSCNN is shown in Fig. 4(b), where the horizontal convolution has a dilation rate 2 (the rectangles in the same color are in one window), and the green and orange arrows denote the horizontal and vertical direction convolutions, respectively. We do not show the vertical convolution for clarity.

Formally, a feature c_i is generated with two-dimensional convolution filter w_c for a token in window $\boldsymbol{x}_{i:i+j-1}$. By sliding the filter window from the beginning of the text T till the end, we get a feature map $\boldsymbol{c}$. After that, we use a max pooling $f_3(\cdot)$ to get one single general text feature:

$$c_i = \sum_{k=1}^{K} \boldsymbol{t}^{wop}[i+rk]w[k] \tag{1}$$

$$\boldsymbol{c} = [c_1, c_2, \ldots, c_{m-j+1}], \tilde{\boldsymbol{h}}_i^{lex} = f(\boldsymbol{c}), \tag{2}$$

where $\boldsymbol{t}^{wop}$ is the input, i is the input index, k is the index in the convolution kernel, r is dilation rate ($r = 1, 2$ denoting the original and dilated CNN for extracting successive and non-successive lexical patterns), $\odot$ represents element-wise product, and $f(\cdot)$ is a non-linear function. The vertical convolution can be defined in a similar way. After performing three different
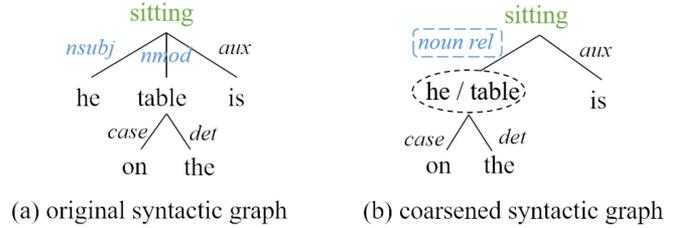
horizontal convolution kernels with the dilation factor r and one vertical convolution kernel, we can obtain the multi-scale lexical feature embedding $\boldsymbol{x}$ for the text T,

$$\boldsymbol{x} = \boldsymbol{h}^{lex} = [\boldsymbol{h}_1^{lex}, \ldots, \boldsymbol{h}_m^{lex}] \tag{3}$$

2) Multi-Scale Syntactic Feature Learning: GCNs are based on the neighborhood aggregation scheme which combines information from neighborhoods to generate node embedding. In our task, one single GCN operation acts as the approximation of aggregation on the first-order neighbors along the dependency path. As shown in Fig. 5(a), 'he,' 'table,' and 'is' can be aggregated to the 'sitting' node, whereas 'on' is the three-hop neighbor of 'he' and thus cannot be aggregated into 'he' in one or two graph convolution operations. Meanwhile, adding too many aggregations will cause the nodes' features over-smoothed and make them indistinguishable.

In view of this, we propose a multi-scale graph convolution network (MSGCN). The key idea in our MSGCN is to form a coarsened graph by merging the nodes with similar syntactic roles into a hypernode, and then perform the graph coarsening and refining operation. Fig. 4(b) shows an example of the coarsened graph where 'he' and 'table' are merged into a hypernode.

a) Graph coarsening and refining operation: Our proposed MSGCN is inspired by H-GCN [52]. However, instead of aggregating structurally similar nodes in H-GCN [52], we merge the nodes having the same relation type with a core node into hypernodes to form the coarsened graph. This enables our MSGCN to learn multi-scale syntactic information by enlarging the receptive field for each word in the hypernode.

Below we present the detail of graph coarsening operation in MSGCN.

The goal in RE tasks is to extract the relation of the entities, and verbs serve as the predicate of a sentence which express an action, an event, or a state. Entities and verbs serve as the most important syntactical functions in sentences. Besides, prepositions show the place, position, time, or method, e.g., 'on the table'. Hence we select the entity, verb, and preposition words as *the core node*, and their neighboring words in the parse tree will be merged into a *hypernode* if they have the same relation type towards the core word.

For example, given the sample 'He is sitting on the table,' 'sitting' is the verb in the sentence and hence is a core node, and both the nsubj relation 'sitting-he' and the nmod relation 'sitting-table' have the same noun relation type. Consequently, 'he' and 'table' are grouped into a hypernode 'he/tabel,' and all their respective neighborhoods are linked to the hypernode. Also note that the relation between 'he/tabel' and 'sitting' ascends to a relation type of 'noun rel'. A new coarsened graph is then formed, as shown in Fig. 4(b). Clearly, with this coarsened graph, the entity word 'he' has a wide receptive field, and we can obtain the relation 'on' between 'he' and 'table' in one convolution operation in MSGCN.

In order to restore the original topological structure of the graph and retain the original semantic information of words, we perform the symmetric graph refining operation, which is done by splitting the hypernode like 'he/tabel' into the original sperate nodes like 'he' and 'table'.

b) Multi-scale GCN: MSGCN is built upon graph convolutional networks (GCN) [61]. At the lth layer, GCN takes the graph adjacency matrix A_l and the hidden representation matrix (h_l^{dep} in our case) as the input, and outputs a hidden representation matrix h_{l+1}^{dep}, which is described as follows:

$$h_{l+1}^{dep} = ReLU(\widetilde{D}_l^{-\frac{1}{2}} \widetilde{A}_l \widetilde{D}_l^{-\frac{1}{2}} h_l^{dep} W_l), \qquad (4)$$

where $h_1^{dep} = t^{wop}$, $\widetilde{A}_l$ is the adjacency matrix with self-loop $\widetilde{A}_l = A_l + I$, $\widetilde{D}_l$ is the degree matrix of $\widetilde{A}_l$, and W_l is a trainable weight matrix.

To learn the multi-scale syntactic features for the word nodes, MSGCN first uses GCN formulated in (4) to obtain the word node representation, and it then applies GCN to the coarsened and refined graphs, respectively. The embedding of the hypernode in the coarsened graph is the summation of the original nodes' embedding, and the graph adjacency matrix is constructed according to the links between the hypernodes and the remaining nodes in the coarsened graph. For the refined graph, the node's embedding is the summation of the hypernode's and the original node's embeddings. Since the graph coarsening and refining operation are symmetric, the graph adjacency matrix in the refined graph equals to that in its corresponding coarsened graph.

By applying the above graph convolution operations to the original, coarsened, and refined graphs, we can finally obtain the multi-scale syntactic feature embedding y as follows:

$$y = h^{dep} = [h_1^{dep}, \ldots, h_m^{dep}] \qquad (5)$$

3) Getting Joint Representations:

a) Multi-scale local representations: By now we get the multi-scale local representations for word sequence and dependency graph, denoted as $h^{lex} = x$ and $h^{dep} = y$ in Fig. 3. We then employ *conv+gate* [62] mechanism to enrich h^{dep} with the help of h^{lex}. Specifically, we apply one-layer convolution operation [20] to x with the convolution filter w_c. We then use the gating mechanism to control the flow of x after convolution towards y, and obtain the enriched multi-scale local representation h^{loc}:

$$g^y = tanh(y + w_c * x), \; \tilde{h}^{loc} = y \odot g^y + y, \qquad (6)$$

where $\odot$ represents the element-wise product, and $*$ represents one layer convolution.

b) Global representations: Since we have got rich local representation from the sentence, we propose to use a Bi-LSTM layer to further obtain global information of the sentence by learning sequential information available in the word sequence. The Bi-LSTM layer takes a series of vectors as inputs, and generates a hidden state vector in the forward and backward directions for each input.

$$h^{glo} = [h_1^{glo}, \ldots, h_m^{glo}] = Bi - LSTM(t_1^{wop}, \ldots, t_m^{wop}), \quad (7)$$

c) Joint representations: With the above local and global representations h^{glo} and h^{loc}, we adopt the bi-attention mechanism [63], [64] to pass information between h^{glo} and h^{loc} according to the attention weights.

Concretely, we first compute the attention score $a_{i,j}$ between the ith global representation and the jth local representation.

$$a_{i,j} = \frac{exp(h_j^{loc} h_i^{glo^T})}{\sum_u^m exp(h_u^{loc} h_i^{glo^T})}, \qquad (8)$$

We next use $a_{i,j}$ to differentiate each word and get a new word embedding $\tilde{h}_i^{glo}$:

$$\tilde{h}_i^{glo} = h_i^{glo} + \sum_j^m a_{i,j} h_j^{glo}, \qquad (9)$$

We can obtain a new local embedding $\tilde{h}_i^{loc}$ in a similar way. Consequently, we obtain refined representations for the global and local representation of sequence, denoted as $\tilde{h}^{glo} = [\tilde{h}_1^{glo}, \ldots, \tilde{h}_m^{glo}]$ and $\tilde{h}^{loc} = [\tilde{h}_1^{loc}, \ldots, \tilde{h}_m^{loc}]$, respectively.

We combine $\tilde{h}^{glo}$ and $\tilde{h}^{loc}$ to form a sentence representation $h = \tilde{h}^{glo} \oplus \tilde{h}^{loc}$. After that, we apply the max pooling operation to h, and then concatenate it with entity representations $e^h{}_1; \ldots; e^h{}_n$, and feed them into a feed-forward neural network to form the final joint representation h', denoted as z in Fig. 3:

$$z = h' = Linear([f(h); e^h{}_1; \ldots; e^h{}_n], \qquad (10)$$

where $f \colon \mathbb{R}^{d \times m} \to \mathbb{R}^d$ is a max pooling function that maps m word vectors to one vector, d is the dimension of the vector h. $e^h{}_i$ indicates the hidden vector corresponding to the i-th entity in h.

Through the multi-scale feature learning module, we obtain two multi-scale local representation x, y, and one multi-scale

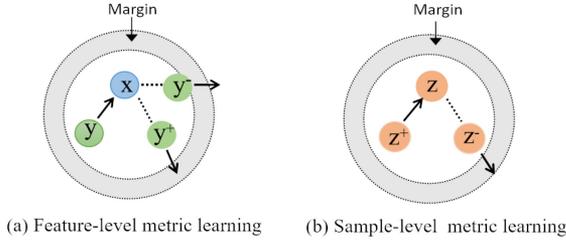(a) Feature-level metric learning   (b) Sample-level metric learning

Fig. 6. Illustration example of our MSML, where 'x' and 'y' are feature embeddings, and 'z' is the relation embedding. '+' and '-' denote the positive and negative sample, and an 'in' and 'out' edge denote the pull and push operation toward the node.

joint relation representation z. These representations will be used in other two modules *MSML* and *MT*.

C. Multi-Scale Metric Learning (MSML) Module

The goal of metric learning is to learn a distance metric that assigns smaller distances between similar pairs and larger distances between dissimilar pairs. Metric learning enables the learned metric propagate the known similarity information to the pairs whose relationships are unknown, which is an essential property and plays a crucial role in many machine learning applications.

In light of this, we present the multi-scale metric learning (MSML) module to assist the RE task, including the feature-level metric learning which is used to reduce/increse the distance between the syntactic and lexical features of the same/different sample(s), and the sample-level metric learning which aims to reduce/increse the distance between the samples in the same/different relation class(es).

The input of the MSML module is the output anchor triplet from the MSFL module, including (x, y, z), (x^+, y^+, z^+), and (x^-, y^-, z^-), where x contains the local multi-scale lexical information, y has the local multi-scale syntactic information, and z is the joint relation representation mainly used for relation classification. '+' and '−' denote the output of positive and negative sample S^+ and 'S^-,' respectively. The triplet (z, z^+, z^-) is used for sample-level metric learning while (x, y, y^+) and (x, y, y^-) are used for feature-level metric learning. Fig. 6 shows an illustration example of our MSML.

When selecting the negative samples, we treat the sample which is most difficult to be distinguished as the negative sample. In particular, we take the anchor S as input and employ the MT module to classify the anchor samples in the validation set. From the probability distribution over relations, the relation with the second highest probability is selected as the negative sample of S.

1) Sample-Level Metric Learning: We first apply the sample-level metric learning to the triplet $[z, z^+, z^-]$ to train our model such that the distance between two samples in the same relation class can be reduced and that from different ones is increased. We use the triple loss as the relation loss function:

$$loss^{rel} = \max(d(z, z^+)^2 - d(z, z^-)^2 + m_1, 0) \quad (11)$$

where $d(\cdot)$ is the Euclidean distance function to measure the similarity between two relation embeddings. $\max(\cdot, 0)$ is the hinge loss and $m_1 > 0$ is the safety margin value of the relation loss function.

2) Feature-Level Metric Learning: In intuition, the local lexical representation x should be close to the local syntactic representation $\tilde{y}$ as they denote the same sentence. However, it is inappropriate to directly let x approximate y since they contain different information. We thus perform feature-level metric learning based on the relative distance. The rationale is that the distance between local pair of the same sample should be smaller than that of different samples. This is done by applying metric learning to the representation triplet (x, y, y^+) and (x, y, y^-). Formally, we define the representation loss as:

$$loss_1^{rep} = \max(d(x, y)^2 - d(x, y^+)^2 + m_2, 0) \quad (12)$$

$$loss_2^{rep} = \max(d(x, y)^2 - d(x, y^-)^2 + m_2, 0) \quad (13)$$

$$loss^{rep} = loss_1^{rep} + loss_2^{rep} \quad (14)$$

where $m_2 > 0$ is the safety margin value of representation loss function.

D. Main Classification (MT) Module

The main task in RE is to determine whether a relation holds for the entities. This task is performed on the joint representation $h' = z$ from the MSFL module. The representation h' is sent into a softmax layer to generate the probability distribution over relations, and the loss for the main relation classification task is defined as follows:

$$\hat{p}^h = softmax(Linear(h')), \tilde{\ } loss^{main} = -p^h log\hat{p}^h, \quad (15)$$

where p^h and $\hat{p}^h$ are the ground truth and the predicted label distribution, respectively.

E. Model Training

Our model is trained with the standard gradient descent algorithm by minimizing the overall loss on all training samples, which consists of the loss for main classification and that for metric learning. Formally, the overall loss is calculated as:

$$loss = \sum_i^J loss_i^{main} + \alpha loss_i^{rel} + \beta loss_i^{rep} + \lambda \|\Theta\|, \quad (16)$$

where J is the number of training samples, α and β are hyper-parameters for controling the weights of $loss^{rel}$ and $loss^{rep}$, respectively. Θ represents all trainable parameters, and λ is the co-efficient of L2-regularization. In the training state, we take the triplet (S, S^+, S^-) as input and in the test state we only use the current sample S as input.

V. EXPERIMENTS

A. Datasets and Settings

1) Datasets: We evaluate our model on two RE tasks including cross-sentence n-ary RE and sentence-level RE on three public datasets.

TABLE I
TRAIN/VALIDATION/TEST SPLITS FOR PUBMED, TACRED AND SEMEVAL

Dataset	PubMed		TACRED		Semeval	
	C	I	C	I	C	I
Train	5	5313	42	68124	10	8000
Validation	5	200	42	22631	-	-
Test	5	1474	42	15509	10	2717

"c" and "i" Denote the Number of Relations and Instances, Respectively.

For cross-sentence n-ary relation extraction, we use the PubMed.[3] dataset [46] which is from the biomedical domain. The dataset contains 6987 ternary instances about drug-gene-mutation relations, and 6087 binary instances about drug-mutation sub-relations. Most instances contain multiple sentences and each instance is assigned with one of the following five labels: "resistance or non-response," "sensitivity," "response," "resistance," and "none". Following [23], [46], we consider two sub-tasks for evaluation, i.e., multi-class n-ary RE and binary-class n-ary RE for which we binarize multi-class labels by grouping all relation classes as "Yes" and treat "None" as "No".

For sentence-level relation extraction task, we follow the experimental settings in [24], [25] to evaluate our model on two datasets. (1) SemEval 2010 Task 8 dataset[4] [65] has 8,000 instances for training and 2,171 instances for testing. It contains 9 directed relations and a special *Other* class. (2) TACRED datase[5] [4] has over 106 k instances and contains 41 relation types and a special *no_relation* class.

Details of train/validation/test splits for PubMed, TACRED, and Semeval are shown in Table I .

2) Settings: For models with static word embedding, we initialize word vectors with the 300-dimension GloVe embeddings provided by [34]. The embeddings for POS and position are initialized randomly and their dimension is set to 30 [24]. We use the Stanford CoreNLP [66] as the parser and SGD as the optimizer with a 0.9 decay rate. The L2-regularization coefficient λ is 0.002.

For models with contextualized word embedding, we use $BERT_{base}$ [32], $BERT_{large}$ [32], RoBERTa [35], and $RoBERTa_{large}$ [35] as pre-trained language models (PLMs), respectively.

For cross-sentence n-ary RE, we use the same data split as that in [23]. Through preliminary experiments on the validation set, we adopt the following combinations of the hyper-parameters (α, β) as they yield the best results: $\alpha = 0.7$ and $\beta = 0.03$ for ternary-relation multi-class, $\alpha = 0.5$, and $\beta = 0.03$ for ternary-relation binary-class, $\alpha = 0.1$ and $\beta = 0.07$ for binary-relation multi-class and $\alpha = 0.1$ and $\beta = 0.09$ for binary-relation binary-class.

For sentence-level RE, following [24], [25], we use the validation set in TACRED and do not use the validation set in

TABLE II
MICRO-AVG. PRECISION, RECALL, AND F1 ON TACRED

Model	P	R	F1
PA-LSTM [4]	65.7	64.5	65.1
Tree-LSTM [67]	66.0	59.2	62.4
LR [4]	73.5	49.9	59.4
SDP-LSTM [45]	66.3	52.7	58.7
C-GCN [25]	69.9	63.3	66.4
C-AGGCN [24]	71.9	64.0	_67.7_
EDCRE [27]	71.2	63.3	67.0
C-GCN-MG [26]	67.5	63.1	65.2
MTB ($BERT_{base}$) [28]	-	-	69.1
MTB ($BERT_{large}$) [28]	-	-	_71.5_
C-GCN-MG ($BERT_{base}$) [26]	68.0	64.4	66.1
CP ($BERT_{base}$) [31]	-	-	69.5
GDPNet ($BERT_{base}$) [70]	71.7	69.0	_70.3_
SP ($BERT_{large}$) [71]	69.8	71.4	70.6
LUKE ($RoBERTa_{large}$) [30]	70.4	75.1	_72.7_
Our model (GloVe)	72.9	64.3	**68.3**†
Our model ($BERT_{base}$)	73.6	67.5	**70.5**
Our model ($BERT_{large}$)	73.4	70.1	**71.7**
Our model ($RoBERTa_{base}$)	71.8	73.7	**72.7**
Our model ($RoBERTa_{large}$)	72.1	74.2	**73.1**

The Marks are as Same as Those in Table IV. Note That There is No Significance Test for PLM Based Models Since the Results for Baselines are Taken From the Original Papers.

TABLE III
MACRO-AVG. F1 ON SEMEVAL

Model	F1
PA-LSTM [4]	82.7
SVM [68]	82.2
SDP-LSTM [45]	83.7
SP-Tree [18]	84.4
C-GCN [25]	84.8
RbSP [69]	85.3
C-AGGCN [24]	85.7
EDCRE [27]	_85.9_
LF-GCN [47]	85.7
C-GCN-MG [26]	82.4
MTB ($BERT_{base}$) [28]	87.3
MTB ($BERT_{large}$) [28]	_89.5_
R-BERT ($BERT_{base}$) [29]	_88.4_
R-BERT ($BERT_{large}$) [29]	89.3
C-GCN-MG ($BERT_{base}$) [26]	85.9
CP ($BERT_{base}$) [31]	87.6
SP ($BERT_{large}$) [71]	89.4
Our model (GloVe)	**86.4**†
Our model ($BERT_{base}$)	**89.1**
Our model ($BERT_{large}$)	**89.8**
Our model ($RoBERTa_{base}$)	**90.3**
Our model ($RoBERTa_{large}$)	**91.8**

The Marks are as Same as Those in Table IV. Note That There is No Significance Test for PLM Based Models Since the Results for Baselines are Taken From the Original Papers.

SemEval but test on the last epoch's checkpoint among 150 training epochs. The best results are based on the combinations of hyper-parameters, i.e., $\alpha = 0.7$ and $\beta = 0.05$ for TACRED and $\alpha = 0.5$ and $\beta = 0.03$ for Semeval.

3) Evaluation Metrics: We use the same metrics as previous work [23]–[25] to evaluate our model. In particular, the test

[3][Online]. Available: https://github.com/freesunshine0316/nary-grn.
[4][Online]. Available: https://github.com/Cartus/AGGCN/tree/master/semeval/dataset.
[5][Online]. Available: https://nlp.stanford.edu/projects/tacred/.

accuracy averaged over five-fold cross validation [23] is used as the metric for the cross-sentence n-ary RE on PubMed, and micro- and macro- averaged F1 scores are used for sentence level RE on TACRED and SemEval dataset, respectively.

B. Results for Sentence-Level RE

1) Baselines: We adopt the popular baselines for sentence-level RE task and group them into three categories.

Sequence based models: We consider the classic PA-LSTM [4] as the representative of sequence based models.

Dependency based models: (1) tree-based methods: the shortest path LSTM (SDP-LSTM) [45], the tree-structure neural model Tree-LSTM [67] and the classic LR classifier over dependency features [4] for TACRED dataset, SVM [68] model combines various linguistic resources for recognizing semantic relations in text. SP-Tree [18] employs bidirectional tree-structured LSTM-RNNs to capture syntactic information on Semeval dataset. RbSP [69] enhances the SDP with the information selected by several attention mechanisms. (2) graph-based methods: C-GCN [24] adopts the pruned trees, C-AGGCN [24] extends the original methods by using contextualized embeddings with a Bi-LSTM as input. EDCRE [27], LF-GCN [47], and C-GCN-MG [26] are same as those in cross-sentence n-ary RE.

PLM based models: MTB [28] is post-trained on Wikidata with randomly masked entity mentions. CP [31] is an entity-masked contrastive post-training framework which also uses Wikidata. MTB [28] and CP [31] both report the results on TACRED and Semeval. LUKE [30] is post-trained to predict randomly masked words and entities in a large entity-annotated corpus retrieved from Wikipedia and it employs RoBERTa$_{large}$ as transformers for TACRED. R-BERT [29] incorporates information from the target entities to tackle the relation classification task on Semeval via fine-tuning. GDPNet [70] constructs a latent multi-view graph to capture various possible relationships among tokens. SP [71] presents a span prediction based system in a fully supervised setup which can force the model to identify the exact entities.

The results for sequence based models and dependency based models are re-implemented by us using GloVe embeddings. The results for these PLM based models are taken from the original papers except that GDPNet [70] and SP [71] are re-produced by us. We re-implement them using the optimal hyper-parameter settings reported in their original papers. Note that, we do not use extra corpus like Wikidata to help relation extraction task in our model. Our proposed post-trained task is conducted on the same RE dataset.

2) Main Results: We present the main results on TACRED and Semeval in Table II and Table III, respectively.

Our model achieves the best results in terms of Micro-F1 on TACRED or Macro-F1 scores on Semeval with GloVe embeddings. Other results are also consistent with those in cross sentence n−ary task.

For PLM based models, we can see that our model beats all the baseline methods, no matter the input contextualized word embedding is from BERT$_{base}$, BERT$_{large}$, RoBERTa$_{base}$,

or RoBERTa$_{large}$. Note that our model with contextualized embeddings gets less improvement over the second best baselines (comparing F1 score BERT$_{large}$ 71.7 *vs.* 71.5, RoBERTa$_{large}$ 73.1 *vs.* 72.7 in Table II). The reason might be that we do not use the extra Wikidata for post-training the word embeddings. Nevertheless, our model can already outperform the baselines without using the extra large corpus. This clearly shows the effectiveness of our model.

C. Results for Cross-Sentence n-Ary RE

1) Baselines: For cross-sentence n-ary RE, we compare our model with following baselines which are grouped into two categories.

Sequence based models: PA-LSTM [4] is a classic model applying LSTM to the word sequence where each word embedding contains word, position, and POS tag.

Dependency based models: We compare our model with three types of dependency based models. (1) Feature-based [72] is a statistical method with features derived from the shortest paths in dependency tree. (2) Tree-based methods: SP-Tree [18] is a tree-structured LSTM method. Graph LSTM-EMBED [46], Graph LSTM-FULL [46], Bidir DAG LSTM [23], and GS GLSTM [23] are LSTM models that adopt graph-structure with dependency as edges. (3) Graph-based methods: GCN [25] pools information over pruned trees. AGGCN [24] takes full dependency trees as inputs with a soft-pruning approach. EDCRE [27] considers both dependency edges between entities and discourse relations between sentences. LF-GCN [47] composes task-specific dependency forests that capture non-local interactions. C-GCN-MG [26] performs GCN operation over multiple dependency sub-graphs and it is the only baseline using BERT to get initial word embeddings.

We re-implement PA-LSTM, EDCRE, and C-GCN-MG (with GloVe embedding) using the optimal hyper-parameter settings reported in corresponding papers since they do not provide full results for all tasks in our experiments. Results for other baselines are taken from the original papers.

2) Main Results: The comparison results for all methods are shown in Table IV. From these results, we make the following observations.

1) Our model achieves the best accuracy in cross-sentence n-ary RE tasks with both GloVe and BERT embeddings. For example, it gets an improvement of 2.1% and 1.7% accuracy over the second best one on ternary and binary relation multi-class RE with GloVe, respectively. Moreover, with the BERT$_{base}$ embedding, our model significantly outperforms its counterpart baseline C-GCN-MG [26] with the same embedding.

2) For baselines with contextualized word embeddings (PLMs), their results significantly outperform the methods with static word embeddings, which is reasonable. In addition, their performances increase with the size of corpus (BERT$_{large}$ > BERT$_{base}$, RoBERTa$_{large}$ > RoBERTa$_{base}$), and they also increase with the performance of PLMs (RoBERTa > BERT). Both these prove the power of PLMs beyond the methods themselves. They also indicate that

TABLE IV
COMPARISON RESULTS IN TERMS OF ACCURACY ON PUBMED

Model	Binary-class				Multi-class	
	T		B		T	B
	Single	Cross	Single	Cross	Crss	Cross
PA-LSTM [4]	84.9	85.8	85.6	85.0	78.1	77.0
Feature-Based [72]	74.7	77.7	73.9	75.2	-	-
SP-Tree [18]	-	-	75.9	75.9	-	-
Graph LSTM-EMBED [46]	76.5	80.6	74.3	76.5	-	-
Graph LSTM-FULL [46]	77.9	80.7	75.6	76.7	-	-
Bidir DAG LSTM [23]	75.6	77.3	76.9	76.4	51.7	50.7
GS GLSTM [23]	80.3	83.2	83.5	83.6	71.1	71.7
GCN (Full Tree) [25]	84.3	84.8	84.2	83.6	77.5	74.3
GCN ($K=0$) [25]	85.8	85.8	82.8	82.7	75.6	72.3
GCN ($K=1$) [25]	85.4	85.7	83.5	83.4	78.1	73.6
GCN ($K=2$) [25]	84.7	85.0	83.8	83.7	77.9	73.1
AGGCN [24]	87.1	87.0	85.2	85.6	79.7	77.4
EDCRE [27]	87.9	_89.0_	86.4	86.8	_82.5_	_79.9_
LF-GCN [47]	_88.0_	88.4	_86.7_	_87.1_	81.5	79.3
C-GCN-MG [26]	87.2	88.5	86.1	86.9	82.1	78.8
C-GCN-MG (BERT$_{base}$) [26]	_88.3_	_88.1_	_87.9_	_87.2_	_86.2_	_87.3_
Our model (GloVe)	**89.1**‡	**89.3**†	**87.7**‡	**87.5**‡	**84.6**‡	**81.6**‡
Our model (BERT$_{base}$)	**90.2**‡	**90.7**‡	**90.5**‡	**89.5**‡	**87.9**‡	**89.1**‡
Our model (BERT$_{large}$)	**91.1**‡	**91.3**‡	**91.6**‡	**91.5**‡	**88.6**‡	**89.6**‡
Our model (RoBERTa$_{base}$)	**91.8**‡	**91.9**‡	**92.4**‡	**92.3**‡	**89.5**‡	**90.4**‡
Our model (RoBERTa$_{large}$)	**92.6**‡	**92.4**‡	**92.8**‡	**92.9**‡	**90.3**‡	**90.8**‡

"t" and "b" Denote the Ternary and Binary Entity Interactions, and "single" and "cross" Mean the Accuracy Calculated Within Single Sentences or on All Sentences. The Best Results With GloVe/PLMs Embeddings are Both in Bold, and the Second Best Ones With GloVe/PLMs Embeddings are underlined/italic. † and ‡ Mark Denote Statistically Significant Improvements Over the Corresponding Second Best Results With $p < .05$ and $p < .01$, Respectively.

a fair comparison between PLM based methods should be conducted using the same PLM, e.g., BERT$_{base}$ $vs.$ BERT$_{base}$.

3) For baselines with static word embeddings, EDCRE and LF-GCN using graphs are better than those using LSTM among dependency based models, indicating that the graph structure is more appropriate for modeling dependency trees than the sequential structure. Moreover, the better performance of LF-GCN can be due to the fact that it develops a new method to obtain task-specific dependency relations. The sequence based model PA-LSTM outperforms the dependency based methods using LSTM and matches those with GCN, indicating that both word sequence and parse tree have their own strengths given the proper architecture to extract lexical and syntactic patterns.

D. Ablation Study

To examine the impacts of each component in our model, we further conduct two types of ablation study. One is for validating the effects of three components (MSCNN, MSGCN, and MSML). This is done by removing the whole component from the model. The other is for validating the effects of multi-scale learning. This is done by replacing MSCNN/MSGCN with the ordinary CNN/GCN. We present the ablation results for sentence-level RE on TACRED and SemEval dataset in

TABLE V
RESULTS FOR ABLATION STUDY(%) ON TACRED AND SEMEVAL DATASET. M1-M6 ARE MODELS WITH GLOVE EMBEDDINGS

Model	TACRED			SemEval
	P	R	Micro-F1	Macro-F1
M1⁻ w/o MSCNN	72.3	63.1	67.4	85.5
M2⁻ w/o MSGCN	72.1	62.8	67.1	85.3
M3⁻ w/o MSML	72.5	63.7	67.8	86.2
M4⁻ r/p MSCNN	72.4	63.4	67.6	86.1
M5⁻ r/p MSGCN	72.5	63.2	67.5	85.9
M6 Our model (GloVe)	72.9	64.3	**68.3**	**86.4**
M7⁻ w/o MSCNN	73.4	67.2	69.7	88.6
M8⁻ w/o MSGCN	73.1	67.3	69.6	88.5
M9⁻ w/o MSML	73.3	67.2	70.3	88.8
M10⁻ r/p MSCNN	73.4	67.1	70.1	88.9
M11⁻ r/p MSGCN	73.2	67.0	70.0	88.8
M12 Our model (BERT$_{base}$)	73.6	67.5	**70.5**	**89.1**

M7-M12 are Models With BERT$_{base}$ Embeddings. w/o Denotes Removing the Whole Component and r/p Denotes Replacing the MSCNN/MSGCN With the Ordinary CNN/GCN.

Table V, and those for for cross-sentence n-ary RE on PubMed in Table VI, respectively.

From the results in Table VI and Table V, we find both the removal and replacement operations incur the performance decrease in comparison with their counterparts, which clearly demonstrates that our three components and the multi-scale learning strategy contribute to our model.

Among three components, the removal of MSGCN results in the biggest performance decrease. For example, with GloVe

TABLE VI
RESULTS FOR ABLATION STUDY(%) ON PUBMED DATASET. M1-M6 ARE MODELS WITH GLOVE EMBEDDINGS

Model	Binary-class				Multi-class	
	T		B		T	B
	Single	Cross	Single	Cross	Cross	Cross
M1⁻ w/o MSCNN	87.8	87.8	86.7	86.0	82.9	79.1
M2⁻ w/o MSGCN	87.3	87.5	86.4	85.8	82.7	79.0
M3⁻ w/o MSML	88.9	89.1	87.4	86.6	84.1	80.9
M4⁻ r/p MSCNN	88.4	88.6	87.4	86.4	83.7	80.8
M5⁻ r/p MSGCN	88.3	88.5	87.2	86.2	83.4	80.6
M6 Our model (GloVe)	**89.1**	**89.3**	**87.7**	**87.5**	**84.6**	**81.6**
M7⁻ w/o MSCNN	89.7	89.6	89.6	88.8	87.3	88.6
M8⁻ w/o MSGCN	89.7	89.7	89.8	88.6	87.1	88.4
M9⁻ w/o MSML	89.8	89.6	89.9	89.1	87.8	88.8
M10⁻ r/p MSCNN	89.9	90.4	90.3	89.2	87.8	88.9
M11⁻ r/p MSGCN	89.8	90.3	90.4	89.4	87.6	88.9
M12 Our model (BERT$_{base}$)	**90.2**	**90.7**	**90.5**	**89.5**	**87.9**	**89.1**

M7-M12 are Models With BERT$_{base}$ Embeddings. w/o Denotes Removing the Whole Component and r/p Denotes Replacing the MSCNN/MSGCN With the Ordinary CNN/GCN.

embeddings, the complete model M6 has an accuracy score 81.6 on PubMed (Multi-class, B) in Table VI. After removing MSGCN, the accuracy for the reduced method M2⁻ drops to 79.0, showing a 1.6 absolute decrease. Similarly, the replacement of MSGCN on the same task also brings about an 1.0 accuracy decrease (from M6 81.6 to M5⁻ 80.6). This clearly proves that our proposed MSGCN enlarges the words' receptive field in the parse tree and shows impressive performance enhancement.

Between the methods with two types of word embeddings, our components have larger positive impacts on the static GloVe embeddings than the contextualized BERT embeddings. For example, on SemEval dataset, the removal of MSCNN for GloVe brings about a 0.9 F1 decrease (from M6 86.4 to M1⁻ 85.5) in Table V. In contrast, the removal of MSCNN for BERT$_{base}$ only has a 0.5 F1 decrease (from M12 89.1 to M1⁻ 88.6). Similar results can be observed for MSGCN and MSML components. The reason might be that BERT has already learned sufficient information from its own multiple transformer layers and the space for improvements is limited, especially on those small datasets like SemEval. Nevertheless, on the larger dataset like TACRED, our multi-scale feature and metric learning paradigm still shows promising improvements.

E. Case Study

To have a close look, we select three examples from three datasets for a case study and present the results of different models in Table VII. We choose PA-LSTM (the classic sequence based model), EDCRE (the best dependency based model) with GloVe embeddings, and C-GCN-MG and R-BERT with BERT$_{base}$ embeddings as the comparison methods for case study.

The first example is from the PubMed dataset. It generally mentions that 'cetuximab' (drug) does not have an effect on 'S492R' (mutation) on 'EGFR' (gene). The context introduces more ambiguity by mentioning that another drug 'panitumumab' is related to 'EGFR' and 'S492R'. Hence both PA-LSTM and EDCRE fail to extract the correct relation among entities. In

TABLE VII
CASE STUDY

Example	Model	Prediction
Panitumumab can still bind to an EGFR mutant S492R to which cetuximab can not bind to.	PA-LSTM$_G$	response ✗
	EDCRE$_G$	response ✗
	Our model$_G$	none ✓
	Our model$_B$	none ✓
But US and Indian experts say it has hesitated to take action against Lashkar-e-Taiba , which means "The Army of the Pure, " believing that the Islamic militants ... rival India.	PA-LSTM$_G$	no_relation ✗
	C-GCN-MG$_B$	member_of ✗
	Our model$_G$	alternate_names ✓
	Our model$_B$	alternate_names ✓
They connected the plug of the battery box to the plug of the charger .	PA-LSTM	Compo-Whole ✗
	R-BERT$_B$	Compo-Whole ✗
	Our model$_G$	Other ✓
	Our model$_B$	Other ✓

The Left Column Presents the Selected Examples, and the Two Columns on the Right Denote the Extraction Results of Corresponding Models. Words in Red are Entities. "g" and "b" Indicate Models With GloVe and BERT$_{base}$ Embeddings.

contrast, our model connects 'cetuximab' with 'not' via the dilated horizontal convolution operation in MSCNN and hence it makes a correct prediction.

The second example is from the TACRED dataset. It aims at detecting the relation 'org:alternate_names' between two organizations 'Lashkar-e-Taiba' (subject) and 'Army of the Pure' (object). Both entities have more than two words, and the sentence is too long for a ordinary CNN and GCN in PA-LSTM or C-GCN-MG$_B$ to capture the relation between two entities. Our model, on the other hand, enlarges the receptive field for each word with the help of MSCNN and MSGCN, and finally our model successfully recognizes the 'org:alternate_names' relation.

The third example is from the SemEval dataset. Its goal is to detect the relation between 'plug' and 'battery box,' which does not exist. PA-LSTM and R-BERT$_B$ treat 'plug' to be a part of 'battery box' and then assign the 'Component-Whole' relation.
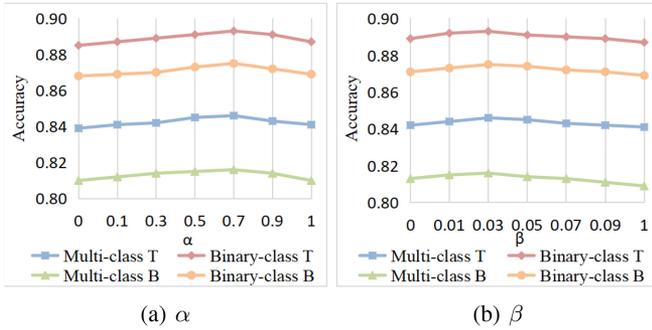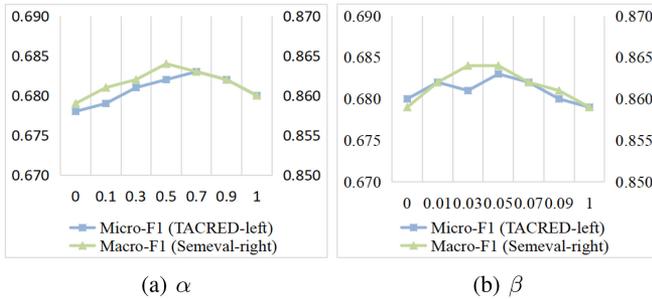
Fig. 7. Impacts of α and β on PubMed.

Fig. 8. Impacts of α and β on TACRED and Semeval.

Our reduced model without MSML also makes the same wrong prediction. However, our complete model can successfully recognize the 'other' relation. This proves the effectiveness of our multi-scale metric learning paradigm.

F. Parameter Analysis

Our model has two key hyper-parameters α and β in Eq. 16, which determine the weights of $loss^{rel}$ and $loss^{rep}$, respectively. This section examines the impacts of α and β on PubMed, TACRED, and Semeval dataset with GloVe embeddings. Because the trend of "Single" is the same with that of "Cross" in binary-class relation extraction on PubMed dataset, we only report the accuracy for "Cross". The results are shown in Fig. 7 and Fig. 8, respectively.

In general, the curves for α and β have similar trends in that they reach a peak first and then gradually decrease. Remember that $loss^{rel}$ and $loss^{rep}$ are used to guide the sample-level and feature-level metric learning. One is for relation label between entities for relation extraction and the other for entity representation. If they are set too small, the model cannot get the guidance from the metric learning module. If they are set too large, the training process inclines towards the metric learning subtask and thus the performance of the main classification decreases.

G. Computational Cost Analysis

We compare the model size (the number of parameters) of our model to four typical baselines with GloVe embeddings on PubMed, TACRED, and Semeval in Table VIII. We use an open source PyTorch implementation[6] to measure the model size for these methods.

[6][Online]. Available: https://github.com/songyouwei/ABSA-PyTorch

TABLE VIII
MODEL SIZE (M) OF OUR MODEL AND THAT OF TYPICAL BASELINES ON PUBMED, TACRED, AND SEMEVAL

| Model | PubMed | | | | TACRED | Semeval |
| | Binary-class | | Multi-class | | | |
	T	B	T	B		
PA-LSTM [4]	1.77	1.83	1.77	1.84	10.8	9.9
AGGCN [24]	1.74	1.64	1.74	1.64	20.4	11.0
C-GCN-MG [26]	5.60	5.54	5.64	5.57	96.4	48.7
EDCRE [27]	5.87	5.86	5.87	5.89	98.5	54.8
Our model	4.18	4.76	4.81	4.32	84.8	40.64

It can be seen that using the same dimension of hidden states, PA-LSTM [4] and AGGCN [24] have a smaller model size compared with other methods, and EDCRE [27] has the biggest cost. Our model is in the middle in terms of the model size but it can produce the best classification results.

VI. CONCLUSION

In this paper, we propose a novel multi-scale feature and metric learning framework for RE tasks. Our methodological contribution is to highlight the importance of building the lexical and syntactic feature and relation hierarchy in text. Our technical contributions are three-fold. Firstly, we design a multi-scale convolution neural network (MSCNN) with dilated horizontal and vertical convolution to extract non-successive lexical patterns from the word sequence. Secondly, we design a multi-scale graph convolution network (MSGCN) to enlarge the receptive field of the word along different syntactic paths. Thirdly, we present a feature-level and sample-level metric learning scheme to further refine the representations.

We conduct extensive experiments on three datasets for cross sentence n-ary and sentence-level RE tasks. The results prove that our model achieves a new state-of-the-art performance with both the static and contextualized word embeddings.

REFERENCES

[1] M. Yu, W. Yin, K. S. Hasan, C. N. dos Santos, B. Xiang, and B. Zhou, "Improved neural relation detection for knowledge base question answering," in *Proc. Assoc. Comput. Linguistics*, 2017, pp. 571–581.
[2] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, "Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3219–3232.
[3] N. Bhutani, X. Zheng, and H. V. Jagadish, "Learning to answer complex questions over knowledge bases with query composition," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 739–748.
[4] 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.

[5] A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi, "COMET: Commonsense transformers for automatic knowledge graph construction," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 4762–4779.

[6] C. D. Sa *et al.*, "Incremental knowledge base construction using deepdive," *Very Large Data Bases J.*, vol. 26, no. 1, pp. 81–105, 2017.

[7] V. Bellini, V. W. Anelli, T. D. Noia, and E. D. Sciascio, "Auto-encoding user ratings via knowledge graphs in recommendation scenarios," in *Proc. 2nd Workshop Deep Learn. Recommender Syst.*, 2017, pp. 60–66.

[8] X. Huang, Q. Fang, S. Qian, J. Sang, Y. Li, and C. Xu, "Explainable interaction-driven user modeling over knowledge graph for sequential recommendation," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 548–556.

[9] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2002, pp. 71–78.

[10] T. Hasegawa, S. Sekine, and R. Grishman, "Discovering relations among named entities from large corpora," in *Proc. Assoc. Comput. Linguistics*, 2004, pp. 415–422.

[11] N. Bach and S. Badaskar, "A review of relation extraction," 2007. [Online]. Available: http://www.cs.cmu.edu/%7Enbach/papers/A-survey-on-Relation-Extraction.pdf

[12] R. T. McDonald, F. C. N. Pereira, S. Kulick, R. S. Winters, Y. Jin, and P. S. White, "Simple algorithms for complex relation extraction with applications to biomedical IE," in *Proc. Assoc. Comput. Linguistics*, 2005, pp. 491–498.

[13] M. Gerber and J. Y. Chai, "Beyond nombank: A study of implicit arguments for nominal predicates," in *Proc. Assoc. Comput. Linguistics*, 2010, pp. 1583–1592.

[14] K. Yoshikawa, S. Riedel, T. Hirao, M. Asahara, and Y. Matsumoto, "Coreference based event-argument relation extraction on biomedical text," *J. Biomed. Semant.*, vol. 2, pp. S6, 2011.

[15] C. Liu, W. Sun, W. Chao, and W. Che, "Convolution neural network for relation extraction," *ADMA, Ser. Lecture Notes Comput. Sci.*, vol. 8347, pp. 231–242, 2013.

[16] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proc. 25th Int. Conf. Comput. Linguistics, Tech. Papers*, 2014, pp. 2335–2344.

[17] L. Wang, Z. Cao, G. de Melo, and Z. Liu, "Relation classification via multi-level attention CNNs," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1298–1307.

[18] M. Miwa and M. Bansal, "End-to-end relation extraction using LSTMs on sequences and tree structures," *ACL*, pp. 1105–1116, 2016.

[19] J. Tourille, O. Ferret, A. Névéol, and X. Tannier, "Neural architecture for temporal relation extraction: A bi-LSTM approach for detecting narrative containers," in *Proc. Assoc. Comput. Linguistics*, 2017, pp. 224–230.

[20] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1746–1751.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2005, pp. 724–731.

[23] L. Song, Y. Zhang, Z. Wang, and D. Gildea, "N-ary relation extraction using graph-state LSTM," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2226–2235.

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

[25] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2205–2215.

[26] A. Mandya, D. Bollegala, and F. Coenen, "Graph convolution over multiple dependency sub-graphs for relation extraction," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 6424–6435.

[27] S. Lee, S. Seo, B. Oh, K. Lee, D. Shin, and Y. Lee, "Cross-sentence N-ary relation extraction using entity link and discourse relation," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 705–714.

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

[29] S. Wu and Y. He, "Enriching pre-trained language model with entity information for relation classification," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 2361–2364.

[30] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto, "LUKE: Deep contextualized entity representations with entity-aware self-attention," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 6442–6454.

[31] H. Peng *et al.*, "Learning from context or names? An empirical study on neural relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process., EMNLP*, 2020, pp. 3661–3672.

[32] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[33] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[34] 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.

[35] Y. Liu *et al.*, "Roberta: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[36] C. Aone and M. Ramos-Santacruz, "REES: A large-scale relation and event extraction system," in *Proc. Int. Conf. Adv. Natural Lang. Process.*, 2000, pp. 76–83.

[37] J. M. Temkin and M. R. Gilder, "Extraction of protein interaction information from unstructured text using a context-free grammar," *Bioinformatics*, vol. 19, no. 16, pp. 2046–2053, 2003.

[38] J. Chen, D. Ji, C. L. Tan, and Z. Niu, "Unsupervised feature selection for relation extraction," *IJCNLP*, pp. 262–267, 2005.

[39] R. C. Bunescu and R. J. Mooney, "Subsequence kernels for relation extraction," in *Proc. Neural Inf. Process. Syst.*, 2005, pp. 171–178.

[40] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2015, pp. 39–48.

[41] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun, "Neural relation extraction with selective attention over instances," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 2124–2133.

[42] D. Zhang and D. Wang, "Relation classification via recurrent neural network," 2015, *arXiv:1508.01006*.

[43] C. Lin, T. A. Miller, D. Dligach, H. Amiri, S. Bethard, and G. Savova, "Self-training improves recurrent neural networks performance for temporal relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 165–176.

[44] P. Zhou *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 207–212.

[45] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, "Classifying relations via long short term memory networks along shortest dependency paths," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1785–1794.

[46] 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, 2017.

[47] Z. Guo, G. Nan, W. Lu, and S. B. Cohen, "Learning latent forests for medical relation extraction," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 3651–3657.

[48] Z. Jiang, W. Xu, J. Araki, and G. Neubig, "Generalizing natural language analysis through span-relation representations," in *Proc. Assoc. Comput. Linguistics*, 2020, pp. 2120–2133.

[49] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.

[50] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Representations*, 2016.

[51] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "HARP: Hierarchical representation learning for networks," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2127–2134.

[52] F. Hu, Y. Zhu, S. Wu, L. Wang, and T. Tan, "Hierarchical graph convolutional networks for semi-supervised node classification," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 4532–4539.

[53] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell, "Distance metric learning with application to clustering with side-information," in *Proc. Neural Inf. Process. Syst.*, 2002, pp. 505–512.

[54] S. Chen, J. L. Moore, D. Turnbull, and T. Joach, "Playlist prediction via metric embedding," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2012, pp. 714–722.

[55] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1701–1708.

[56] J. Wan *et al.*, "Deep learning for content-based image retrieval: A comprehensive study," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 157–166.

[57] M. Köstinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof, "Large scale metric learning from equivalence constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2288–2295.

[58] Z. E. Xu, M. Chen, K. Q. Weinberger, and F. Sha, "From sBoW to dCoT marginalized encoders for text representation," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2012, pp. 1879–1884.

[59] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.

[60] C. Hsieh, L. Yang, Y. Cui, T. Lin, S. J. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. Int. Conf. World Wide Web*, 2017, pp. 193–201.

[61] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[62] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 933–941.

[63] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 289–297.

[64] P. Gao *et al.*, "Dynamic fusion with intra- and inter-modality attention flow for visual question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6639–6648.

[65] I. Hendrickx *et al.*, "Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," in *Proc. Assoc. Comput. Linguistics*, 2010, pp. 33–38.

[66] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. Assoc. Comput. Linguistics*, 2014, pp. 55–60.

[67] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proc. Assoc. Comput. Linguistics*, 2015, pp. 1556–1566.

[68] B. Rink and S. M. Harabagiu, "UTD: Classifying semantic relations by combining lexical and semantic resources," in *Proc. Assoc. Comput. Linguistics*, 2010, pp. 256–259.

[69] D. Can, H. Le, Q. Ha, and N. Collier, "A richer-but-smarter shortest dependency path with attentive augmentation for relation extraction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 2902–2912.

[70] F. Xue, A. Sun, H. Zhang, and E. S. Chng, "GDPNet: Refining latent multi-view graph for relation extraction," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 14 194–14 202.

[71] A. D. N. Cohen, S. Rosenman, and Y. Goldberg, "Relation extraction as two-way span-prediction," 2020, *arXiv:2010.04829*.

[72] 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.

Mi Zhang (Student Member, IEEE) received the B.S. degree in software engineering from Shandong University, Jinan, China. She is currently working toward the master's degree with the School of Computer Science, Wuhan University, Wuhan, China. She has authored or coauthored several papers in leading conferences, such as EMNLP and DASFAA. Her research interests include information extraction and sentiment analysis.
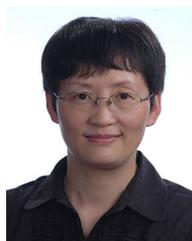
Tieyun Qian (Member, IEEE) received the B.S. degree in computer science from the Wuhan University of Technology, Wuhan, China, in 1991, and the Ph.D. degree in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2006. She is currently a Professor with the School of Computer Science, Wuhan University, Wuhan, China. She has authored or coauthored more than 80 papers in leading conferences and top journals, including ACL, SIGIR, AAAI, EMNLP, CIKM, TKDE, TOIS, and TKDD. Her research interests include web mining and natural language processing. She is a Member of ACM CCF. She was a Senior Program Committee Member of many premium conferences, such as WWW, ACL, EMNLP, AAAI, IJCAI, ICDM, and CIKM.

Bing Liu (Fellow, IEEE) received the Ph.D. degree in artificial intelligence from The University of Edinburgh, Edinburgh, U.K. He is currently a Distinguished Professor of computer science with the University of Illinois at Chicago, Chicago, IL, USA. He has authored or coauthored extensively in top conferences and journals, and authored four books: one about lifelong learning, two about sentiment analysis, and one about web mining. His research interests include lifelong and continual learning, sentiment analysis, lifelong learning chatbots, open-world AI/learning, natural language processing, and data mining and machine learning. From 2013–2017, he was the Chair of ACM SIGKDD, a Program Chair of many leading data mining conferences. He is a Fellow of the ACM and AAAI.