Graph Convolution over Multiple Dependency Sub-graphs for Relation Extraction

Angrosh Mandya, Danushka Bollegala* and Frans Coenen
Department of Computer Science, University of Liverpool, UK
{angrosh,danushka.bollegala,coenen}@liverpool.ac.uk

Abstract

We propose in this paper a contextualised graph convolution network over multiple dependency sub-graphs for relation extraction. A novel method to construct multiple sub-graphs using words in shortest dependency path and words linked to entities in the dependency graph is proposed. Graph convolution operation is performed over the resulting multiple sub-graphs to obtain more informative features useful for relation extraction. Our experimental results show that the proposed method achieves superior performance over existing GCN-based models achieving state-of-the-art performance on cross-sentence n-ary relation extraction and SemEval 2010 Task 8 sentence-level relation extraction task. Our model also achieves a comparable performance to the SoTA on the TACRED dataset.

1 Introduction

In recent times, Graph Convolutional Network (GCN) (Kipf and Welling, 2016) based models such as Contextualised GCN (C-GCN) (Zhang et al., 2018) and Attention Guided GCN (AGGCN) (Guo et al., 2019) are shown to be useful for relation extraction. The C-GCN model employs pruned dependency trees obtained using a *path-centric pruning distance* to filter irrelevant nodes from sentence graph to aid in relation extraction. Experiments with C-GCN model shows that the performance of the C-GCN model significantly decreases with the inclusion of all nodes in the sentence graph obtained from the dependency graph (Zhang et al., 2018), thus making it important to use a right pruning distance to achieve optimum performance. The AGGCN model (Guo et al., 2019) on the other hand, instead of using pruned dependency trees, considers the full dependency tree with a self-attention mechanism to learn node representation useful for relation extraction. In contrast to these GCN-based models using a single graph, we propose in this paper to use multiple sub-graphs constructed from dependency tree to represent a sentence for relation extraction. Given the decrease in performance of C-GCN with an increase in the size of graph (Zhang et al., 2018), we hypothesise that using multiple sub-graphs in place of a single graph can boost performance of GCN-based models for relation extraction. In comparison to the AGGCN model (Guo et al., 2019), our proposed model simply considers smaller graphs associated with entities to learn representation for important nodes in the sentence.

Further, while the pruning strategy used by the C-GCN model (Zhang et al., 2018) is applicable for sentence-level relation extraction, it is not suitable for cross-sentence n-ary relation extraction as there could be more than two entities across multiple sentences. The challenges arise in obtaining a single dependency path in the LCA sub-tree connecting n-ary entities across sentences for cross-sentence relation extraction. Previous work on cross-sentence relation extraction task (Peng et al., 2017; Song et al., 2018; Guo et al., 2019) have largely made use of full dependency graph with additional features such as adjacent words and co-reference links (Peng et al., 2017). The usage of full dependency tree significantly increases network parameters, requiring high computational power. In contrast to these works on n-ary relation extraction, we propose in this paper a novel method to construct sub-graphs using connections

*Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

between entities and root terms in dependency tree across sentences. Our approach described in this paper significantly reduces the number of nodes in a graph and also at the same time helps in achieving higher performance for cross-sentence n-ary relation extraction.

More specifically, the key contributions of this paper are: (a) propose a C-GCN model over multiple sub-graphs (C-GCN-MG) for cross-sentence n-ary relation extraction and sentence-level relation extraction tasks; (b) propose novel methods to construct sub-graphs around entities using the dependency tree for relation extraction; (c) provide evidence to substantiate the use of multiple sub-graphs instead of a single graph for relation extraction; and (d) evaluate C-GCN-MG model on standard datasets for relation extraction and show that C-GCN-MG achieves SoTA performance on cross-sentence n-ary relation extraction dataset (Peng et al., 2017) and SemEval 2010 Task 8 dataset (Hendrickx et al., 2019) to outperform C-GCN and AGGCN models. We also show that C-GCN-MG achieves comparable performance against C-GCN and AGGCN models on TACRED dataset (Zhang et al., 2017).

2 Related Work

GCNs have been successfully applied for various information retrieval tasks in different areas such as bioinformatics (Borgwardt et al., 2005), chemoinformatics (Duvenaud et al., 2015), social network analysis (Backstrom and Leskovec, 2011), urban computing (Bao et al., 2017) and natural language processing (Borgwardt et al., 2005). With specific application to relation extraction, several studies have examined the use of graph-based mechanisms to improve relation extraction. For example, a combination of dependency graphs, PropBank and FrameNet based features and surface-level lexical features were successfully used with support vector machines to improve relation extraction (Rink and Harabagiu, 2010). Sequence-based Neural network models such as Recurrent Neural Networks (RNNs), LSTMs, BiLSTMs have been used with graph-based features for relation extraction (Socher et al., 2012). Shortest dependency path (SDP) between entities combined with RNNs are also shown to be useful for relation extraction (Xu et al., 2015). Further, word sequence and dependency tree substructures are combined to jointly model entity and relation extraction (Miwa and Bansal, 2016). More recently, graph-based LSTM networks have been shown particularly useful in the context of n-ary cross-sentence relation extraction (Peng et al., 2017; Song et al., 2018). GCN-based models such C-GCN (Zhang et al., 2018) and AGGCN (Guo et al., 2019) are shown to achieve SoTA performance on sentence-level and cross-sentence relation extraction. A combined neural network model that brings together the usefulness of LSTM networks in learning from longer sequences and CNNs to capture salient features is also proposed cross-sentence n-ary relation extraction (Mandya et al., 2018). The main focus of this paper is to examine the use of multiple sub-graphs along with graph convolution models for relation extraction. In contrast to previous GCN-based models for relation extraction such as C-GCN (Zhang et al., 2018) and AGGCN (Guo et al., 2019) which use a single graph to learn node representations, we propose in this paper a C-GCN model that uses multiple sub-graphs to learn a richer node representation that helps in relation extraction. For this purpose, we propose a novel method to obtain multiple sub-graphs from dependency parse trees of a given sentence. To the best knowledge of the authors, this is the first study which effectively combines GCNs with multiple sub-graphs for relation extraction and shows that such a strategy is useful for achieving SoTA performance for relation extraction.

3 Constructing multiple sub-graphs for relation extraction

We propose a method to encode multiple sub-graphs with GCNs for two different tasks: (a) sentence-level relation extraction; and (b) cross-sentence n-ary relation extraction as follows:

3.1 Sentence-level relation extraction

In sentence-level relation extraction, the task is to identify binary relation across entities e_1 and e_2 in a given sentence. For instance, for EXAMPLE SENTENCE 1, given below, the task is to identify the relation *Entity-Origin* across entities *knowledge* (e_1) and *recruits* (e_2).

EXAMPLE SENTENCE 1: "Their **knowledge** of the power and rank symbols of the Continental empires was gained from the numerous Germanic **recruits** in the Roman army, and from the Roman practice of various Germanic warrior groups with land in the imperial provinces."

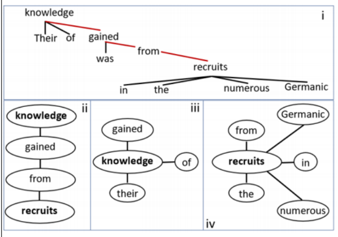

Figure 1: Multiple graphs constructed to represent Example Sentence 1

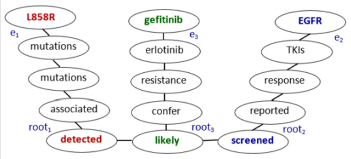

Figure 2: Graph constructed using nodes in SDP connecting 3 entity terms (ternary relation) and root terms in the dependency tree of each sentence.

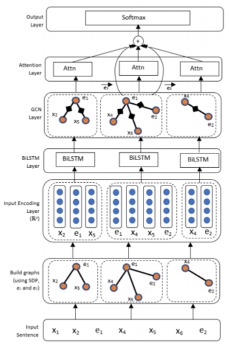

Figure 3: Multiple graphs based GCN for Relation Extraction

Accordingly, multiple sub-graphs (as shown in Figure 1) are constructed to represent a sentence, which includes (a) graph using nodes in SDP between two entities "knowledge" and "recruits" (Figure 1(ii)); and (b) graph using nodes linked to entity mentions "knowledge" (e_1) (Figure 1(iii)) and entity mention "recruits" (e_2) (Figure 1(iv)). The SDP is defined as the dependency path in the LCA sub-tree, connecting the two entities. Figure 1(i) shows partial the dependency tree for EXAMPLE SENTENCE 1.

3.2 Cross-sentence n-ary relation extraction

While the above described method to construct multiple graphs is applicable for sentence-level relation extraction, it cannot be used in the context of cross-sentence n-ary relation extraction, where the task is to identify the n-ary relation across n-ary entities present across multiple sentences. For instance, the EXAMPLE TEXT 1 provided below is an instance of cross-sentence n-ary relation extraction, where the task is to identify *ternary* relation *sensitivity* across three entities **L858R, EFGR, gefitnib** present in the first, second and third sentence, respectively.

EXAMPLE TEXT 1: "Furthermore, although common mutations , such as exon 19 deletions and **L858R** mutations in exon 21 have been associated with response to EGFR TKIs, many other mutations are detected only occasionally , and correlations with response are not defined . A recent study screened 681 cases and found 18 rare mutations; responses to **EGFR** TKIs were reported on a case by case basis and varied by mutation. For example, exon 20 and 21 mutations were more likely to confer resistance to erlotinib or **gefitnib**, while exon 18 and 19 mutations were more often associated with improved efficacy outcome."

To build graph connecting multiple entities, we propose a two-fold strategy. We first use nodes in SDP between each entity term and the root term in the dependency tree to derive graphs around each entity terms, followed by adding an edge between each entity terms to obtain a full-connected graph. For example, in Figure 2, the nodes in SDP between entity terms "L858R", "EGFR" and "gefitnib" and their respective root terms "detected", "likely" and "screened" in the dependency tree of sentence 1, 2 and 3, respectively are used to create the initial graphs for three entities (entity mentions and root terms present in different sentences are shown in different colours in Figure 2), followed by adding an edge between entities. This strategy helps to build a fully-connected graph irrespective of the number of entities and sentences in the text. In addition, we also construct two sub-graphs using nodes associated with the first and the last entity in the cross-sentence instance, as described previously for sentence-level relation extraction. This method of deriving multiple sub-graphs for both cross-sentence n-ary relation instances, results in constructing three sub-graphs to represent each instance, irrespective of the number of entities and sentences present in the instance.

4 C-GCN over Multiple Sub-graphs

In this section, we formally describe the problem and explain the architecture of C-GCN-MG model.

4.1 Problem Formulation

Let $\mathcal{E} = [e_1, .., e_n]$ be the set of entities in a text span $s_i \in \mathcal{S}$ containing t consecutive sentences. $\mathcal{S}$ is the set of all relation extraction instances. Given $\mathcal{E}$ and s_i, the relation extraction task is to predict a relation

r from a predefined set $\mathcal{R}$ that holds across entities $[e_1, .., e_n]$ or "no relation" otherwise. In this work, we apply C-GCN-MG in two different settings: (a) when n=2 and t=1, we predict binary relation in a single sentence which forms sentence level relation extraction problem; and (b) when $n \geq 2$ and $t > 1$, we predict n-ary relation across t sentences also known as cross-sentence n-ary relation extraction problem. In both the settings, we model the task of relation extraction as *graph classification* problem. Each instance s_i is transformed into set of sub-graphs, upon which GCN operation is performed to learn rich node representations in each sub-graph. Using an attention mechanism, each sub-graph is transformed into a fixed dimensional vector. The attentions vectors resulting from multiple sub-graphs combined to obtain entity-centric feature vector, which is used to predict relation r for the instance s_i.

4.2 Architecture of C-GCN-MG

The architecture of the proposed C-GCN-MG model shown in Figure 3 is described below.

4.2.1 Graph Building Layer

The input to the network is the sequence of tokens $[x_1, \ldots, x_m] \in s_i$, which also consists entities $[e_1, \ldots, e_n]$ across which holds a relation $r \in \mathcal{R}$. For explanation purposes, in Figure 3, the input s_i provided to the network is a single sentence ($t = 1$) comprising entities e_1 and e_2. Initially, s_i is transformed into a set of 3 sub-graphs, following the method described in Section 3, as shown in Figure 3. A set of 3 graphs are constructed for each instance irrespective of whether the input is a sentence-level binary or cross-sentence n-ary relation instance to obtain graph $\mathcal{G}(s_i) = g_i^k$, where $k = 1, 2, 3$. An adjacency matrix ($\mathbf{A}$) is obtained for each graph that provides information of connected nodes in the graph. An edge weight matrix ($\mathbf{E}$) providing weight of the grammatical relation for edges between nodes is also obtained. The method for obtaining edge weights is further explained in section 4.2.4.

4.2.2 Input Encoding Layer

Each graph g_i^k comprise a set of tokens $[x_1^k, \ldots, x_{m'}^k]$, which also forms the nodes in the graph is encoded into a fixed-length vector by the input coding layer comprising the following embeddings: (a) contextual; (b) part-of-speech; (c) dependency; (d) named entity type; and (e) word type embeddings. For contextual embeddings, the $\mathbf{BERT}$ model (Devlin et al., 2018) is used. Byte-pair-encoding (BPE) tokeniser used by $\mathbf{BERT}$ tokenises each a word w into s BPE tokens $w = \{b_1, b_2, ..., b_s\}$ and generates L hidden states for each BPE token, $\mathbf{h}_t^l, 1 \leq l \leq L, 1 \leq t \leq s$. The contextual embedding $\mathbf{BERT}_w$ for word w is obtained by summing the last four layers of the BERT model:

$$\mathbf{BERT}_w = \frac{1}{s} \sum_{l=L-4}^{L} \sum_{t=1}^{s} \mathbf{h}_t^l \tag{1}$$

The best performance was observed when the last four layers of the $\mathbf{BERT}$ model were used in the experiments. In addition to contextual embeddings, for each word w, a p-dimensional feature vector is included for Part-of-Speech (POS) tags (f_w^{pos}); dependency grammatical relations (f_w^{dep}); and named entity types (f_w^{net}) embeddings. Further, a q-dimensional feature vector is included to indicate whether a given word w is the entity mention or not (f_w^{wt}). The syntactic embeddings are randomly initialised. Word-type embeddings comprise of a q-dimensional vector of ones, if the given word is an entity mention or vector of zeros if the word is a non-entity term. Thus, the input vector for each token $\tilde{x}_i^k \in \mathbf{R}^d$ and is:

$$\tilde{x}_i^k = [\mathbf{BERT}_{x_i^k}; f_{x_i^k}^{pos}; f_{x_i^k}^{dep}; f_{x_i^k}^{net}; f_{x_i^k}^{wt}] \tag{2}$$

4.2.3 BiLSTM Layer

Previously, using a BiLSTM layer is found to be useful for fine-tuning pre-trained input word embeddings before providing them as input to GCN layer (Zhang et al., 2018; Guo et al., 2019). Although BERT provides contextual embeddings, we propose to use a BiLSTM contextual layer to further fine-tune input embeddings by learning sequential information available in the word order in the sentence. Accordingly, the encoded set of tokens $[\tilde{x}_1^k, \ldots, \tilde{x}_{m'}^k]$ in the graph are initially fed as input to a contextual

BiLSTM layer. As seen in Figure 3, the word sequence for terms in graph is obtained from the sentence and is provided as input to the BiLSTM layer, as experiments using full sequence showed a decrease in performance. Thus, the BiLSTM layer takes as input a series of input vectors and produces a d_l-dimensional hidden state vector for each input in both forward and backward directions. The BiLSTM layer is jointly trained along with the rest of the model. The output of the BiLSTM layer is given by:

$$\mathbf{h}_1^k, \ldots, \mathbf{h}_{m'}^k = \mathbf{BiLSTM}(\tilde{x}_1^k, \ldots, \tilde{x}_{m'}^k) \tag{3}$$

Here, $\mathbf{h}_i \in \mathbb{R}^{2d_l}$, where d_l is the dimension of the hidden state of the LSTM. $\mathbf{h}_i$ is the hidden state vector of the BiLSTM at time-step i considering both forward and backward directions.

4.2.4 Graph Convolution (GCN) Layer

The hidden state vectors obtained from the BiLSTM layer and the graphical structure obtained in the first step (*Graph Building Layer*) is provided as input to the GCN layer. Specifically, for graph g_i^k, the GCN takes the following inputs: (a) an input feature matrix $\mathbf{X} \in \mathbb{R}^{n \times 2d_l}$, where n is the number of nodes and $2d_l$ is the dimensions of input features (output obtained from BiLSTM layer for each node); (b) the graph structure is provided by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $\mathbf{A}_{ij} = 1$, if there exists an edge from node i to node j; and (c) an edge weight vector $\mathbf{e} \in \mathbb{R}^{2n_e}$, where n_e is the number of edges in the graph. The dimension is $2n_e$ as the edge vector for a given pair of nodes is considered in both directions. The edge weights are obtained as follows:

Let $e_{p_i,q_j}^{r_g}$ be the edge weight of the grammatical relation $r_g \in \mathcal{R}_g$ going from node i to node j. $\mathcal{R}_g$ is the set of all grammatical relations; p and q are the POS tags of nodes i and j, respectively. Both p and q are in set P, where P is the set of POS tags seen in the dataset and $p == q$ for nodes i and j with the same POS tag. For a given $p, q \in P$ and $r_g \in \mathcal{R}_g$, if $pq_{n'}$ is the number of times the triples (p_i, q_j, r_g) is seen in the corpus and $pq_{t'}$ is the total number of triples across all POS tags and grammatical relations, the edge weight from node i to node j with a given dependency relation r_g is given by: $e_{p_i,q_j}^{r_g} = \frac{pq_{n'}}{pq_{t'}}$

Thus, the graph convolution operation to produce node features $h_i^l \in \mathbb{R}^{d_g}$ at layer l is given by:

$$h_i^l = \sigma(\sum_{j=1}^{n} \mathbf{A}_{ij} e_{p_i,q_j}^{r_g} \mathbf{W}^{(l)} h_j^{(l-1)} + \boldsymbol{b}^{(l)}) \tag{4}$$

where $\mathbf{W}^{(l)}$ and $b^{(l)}$ is the weight matrix and bias term, respectively for the l^{th} layer, h^{l-1} are the node features in the $l - 1^{th}$ layer, the initial layer $\mathbf{H}^{(0)} = \mathbf{X}$, and σ is the non-linear function such as $\mathbf{ReLU}$, d_g is the dimensions of the hidden state in GCN layer. Although, the graph convolution operation helps to obtain node features h_i, the resulting features do not provide a complete representation as the features of its own node are not considered by the convolution operation. To address this problem, a self-loop is added for each node in the graph (Kipf and Welling, 2016). Further, since certain nodes in the dependency graph can be high-degree nodes with higher connections, the node representation obtained from is likely to favour high-degree nodes, resulting in bias in overall sentence representation. To solve this issue, the node features are normalised by transforming the adjacency matrix $\mathbf{A}$, where $\mathbf{A}$ is multiplied with the inverse degree matrix $\mathbf{D}$ (Kipf and Welling, 2016). Applying these two transformations, the graph convolution operation to produce node features h_i at layer l is given by:

$$h_i^l = \sigma(\sum_{j=1}^{n} \tilde{A}_{ij} e_{p_i,q_j}^{r_g} \mathbf{W}^{(l)} h_j^{(l-1)} / d_i + b^{(l)}) \tag{5}$$

Here, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ with $\mathbf{I}$ being the $n \times n$ identity matrix, and $d_i = \sum_{j=i}^{n} \tilde{A}_{ij}$ is the degree of token i in the dependency graph. The output of graph convolution operation is the node level output $\mathbf{Z} \in \mathbb{R}^{m' \times f}$, where m' is the number of nodes in the graph and f is the number of output features of the GCN layer. Intuitively, the feature representation of node $h_i^L \in \mathbb{R}^f$ is an aggregation of information from the connecting neighbouring nodes and edges in the graph.

4.2.5 Attention Layer

Not all nodes in the GCN output would equally contribute to relation extraction. An attention layer is used to obtain a fixed length vector that best represents the graph. Thus, instead of using multiple GCN layers, a single GCN layer followed by an attention is layer is employed for this purpose. This also significantly reduce the number of parameters in the model. The attention mechanism assigns a weight α_i to each node annotation h_i. A fixed representation $v_{g_i} \in \mathbb{R}^{d_g}$ is computed for the entire graph, as the weighted sum of all node annotations:

$$u_i = \tanh(Wh_i + b) \tag{6}$$

$$\alpha_i = \frac{\exp(u_i)}{\sum_{t=1}^{T} \exp(u_t)}, \quad \sum_{i=1}^{T} \alpha_i = 1 \tag{7}$$

$$v_{g_i} = \sum_{i=1}^{T} \alpha_i h_i, \quad v_{g_i} \in \mathbb{R}^{d_g} \tag{8}$$

The final representation v for the input sequence is obtained by summing all three attention vectors (from three sub-graphs) along with hidden state vectors of entity mentions e_1 and e_2 obtained at GCN layer:

$$v = \sum_{i=1}^{2} h^l_{e_i} + \sum_{k=1}^{3} v_{g^k_i}, \quad h^l_{e_i} \in \mathbb{R}^{d_g},\ v_{g^k_i} \in \mathbb{R}^{d_g} \tag{9}$$

where $h^l_{e_1}$ and $h^l_{e_2}$ are hidden state vectors of entity mentions e_1 and e_2 at layer l of GCN, respectively.

4.2.6 Output Layer

The final feature vector $v \in \mathbb{R}^{d_g}$ is used for classification and is fed to a fully connected softmax layer to obtain a probability distribution over relation labels. The cross-entropy loss for label prediction is given by:

$$J(\theta) = \sum_{i=1}^{r} \log p(r_q|v, \theta), \tag{10}$$

where r is the total number of relations and θ are the parameters of the model. During inference, the test sentences are represented as graphs and fed to the classifier model to predict the relation label.

5 Experiments

5.1 Datasets and Metrics

The performance of our model is evaluated on two tasks: cross-sentence n-ary relation extraction and sentence-level relation extraction. For cross-sentence n-ary relation extraction task, we use the dataset introduced by (Peng et al., 2017) (n-ary dataset), which contains 6,987 ternary relation instances and 6,087 binary relation instances extracted from PubMed[1]. For sentence level relation extraction task, the performance of our model is evaluated on two datasets: (a) SemEval-2010 Task 8 (SemEval) dataset (Hendrickx et al., 2019); and (b) TACRED dataset (Zhang et al., 2017). SemEval is a standard dataset for relation extraction containing 10,717 examples annotated with 9 different relation types, and an artificial relation ‘Other’. The dataset is split into 8,000 training examples and 2,717 test examples, with each sentence marked with two nominals, e_1 and e_2. The TACRED dataset is a larger dataset comprising 106K instances annotated using 41 relation types and a special “no relation” types. Models are evaluated using previously used metrics. For the cross-sentence n-ary relation extraction task, test accuracies averaged

[1]The dataset is available at https://github.com/freesunshine0316/nary-grn

over five cross-validation folds are reported following previous evaluation methods (Peng et al., 2017; Song et al., 2018; Guo et al., 2019). For sentence-level relation extraction, we report the official macro F1-Score excluding the ‘Other’ relation for SemEval dataset (Hendrickx et al., 2019) and for TACRED dataset we report official micro F1-scores (Zhang et al., 2017).

5.2 Implementation Details

PyTorch (Paszke et al., 2017) and PyTorch Geometric (PyG) (Fey and Lenssen, 2019) was used to build the GCN-based model. Spacy (Honnibal and Montani, 2017) was used to obtain POS tags, named entity types, and dependency relations. Since SemEval dataset has dedicated train and test sets, 10% of the training dataset was held-out for validation purposes. The hyperparameters of the model were tuned using the validation set. The model was developed using the validation set and was tested on the test set. The model was trained for 200 iterations following mini-batch gradient descent (SGD) with a batch size of 50. Word embeddings were initialised using 768-dimensional contextual BERT embeddings. The dimensions for embeddings for part-of-speech (POS), named entity tags, dependency tags was set to 40 and were initialised randomly. The dimensions for word-type embeddings was set to 10. A vector of size 10 with all entries as ones and zeros was added for entity mentions and non-entity words, respectively. The dimensions of hidden state vector in the LSTM, GCN and Attention layer was set to 256.

6 Results and Discussion

6.1 Cross-Sentence n-ary Relation Extraction

For cross-sentence n-ary relation extraction, following prior work (Zhang et al., 2018; Guo et al., 2019), the proposed model is evaluated against the following baselines: (1) feature-based classifier using lexical features in the SDP between each pair of entities (Quirk and Poon, 2016); (2) graph-based LSTMs including (a) Graph-LSTM (G-LSTM) (Peng et al., 2017); (b) Bidirectional Directed Acyclic Graph LSTM (Bidir DAG LSTM) (Song et al., 2018); and (c) Graph State LSTM (GS-LSTM) (Song et al., 2018); (3) GCN-based models including (a) Contextualised GCN (C-GCN) (Zhang et al., 2018); and (b) Attention Guided GCN (AGGCN) (Guo et al., 2019); and (4) in addition, following (Song et al., 2018), the tree-structured LSTM model (SPTree) (Miwa and Bansal, 2016) is also included as a baseline for drug-mutation binary relation extraction.

The five-fold cross validation results of the proposed model are provided in Table 1. For ternary relation extraction (first two columns in Table 1), the proposed C-GCN-MG model achieves an accuracy of 88.3 and 88.1, on instances within single sentences and on all instances, respectively, outperforming all baselines. For binary relation extraction (third and fourth columns in Table 1), the C-GCN-MG model consistently outperforms both GS G-LSTM and AGGCN model, indicating the usefulness of the proposed model. Following prior work (Song et al., 2018; Guo et al., 2019), the proposed model is evaluated on all instances for both ternary and binary relations (last two columns in Table 1) for multi-class classification. As seen, the performance of different model significantly drops when evaluated on fine-grained classes against binary class. However the C-GCN-MG model performs better on fine-grained classes models by scoring higher than SoTA models for ternary and binary multi-class relation extraction. The binary and multi-class n-ary results clearly establish the ability of C-GCN-MG to exploit the underlying graph structure by using multiple sub-graphs. While, the performance of GCN is observed to decrease with the increase in nodes (Zhang et al., 2018), the result obtained by C-GCN-MG model, particularly on multi-class classification (Table 1) show that the use of smaller sub-graphs considerably helps fine-grained classification.

6.2 Sentence-level Relation Extraction

Results on Semeval 2010 Task 8 and TACRED datasets are reported for sentence-level relation extraction task. Following prior work (Zhang et al., 2018; Guo et al., 2019), we use the following baselines: (a) dependency-based models: which include logistic regression (LR) (Zhang et al., 2018); Shortest Path LSTM (SDP-LSTM) (Xu et al., 2015), tree-structured neural model (Tree-LSTM) (Tai et al., 2015), GCN and Contextualized GCN (C-GCN) (Zhang et al., 2018), Attention guided GCN (AGGCN) (Guo et al.,

	Binary-class				Multi-class	
	T		B		T	B
Model	S	C	S	C	C	C
Feature-Based (Quirk and Poon, 2016)	74.7	77.7	73.9	75.2	-	-
SPTree (Miwa and Bansal, 2016)	-	-	75.9	75.9	-	-
G-LSTM-EMBED (Peng et al., 2017)	76.5	80.6	74.3	76.5	-	-
G-LSTM-FULL (Peng et al., 2017)	77.9	80.7	75.6	76.7	-	-
+multi-task (Peng et al., 2017)	-	82.0	-	78.5	-	-
GS G-LSTM (Song et al., 2018)	80.3	83.2	83.5	83.6	71.7	71.7
Bidir DAG LSTM (Song et al., 2018)	75.6	77.3	76.9	76.4	51.7	50.7
GCN (Full Tree) (Zhang et al., 2018)*	84.3	84.8	84.2	83.6	77.5	74.3
GCN (K=0) (Zhang et al., 2018)*	85.8	85.8	82.8	82.7	75.6	72.3
GCN (K=1) (Zhang et al., 2018)*	85.4	85.7	83.5	83.4	78.1	73.6
GCN (K=2) (Zhang et al., 2018)*	84.7	85.0	83.8	83.7	77.9	73.1
AGGCN (Guo et al., 2019)	87.1	87.0	85.2	85.6	79.7	77.4
C-GCN-MG (our model)	**88.3**	**88.1**	**87.9**	**87.2**	**86.2**	**87.3**

Table 1: Average test accuracies in five-fold validation for binary-class and multi-class n-ary relation extraction. “T” and “B” denotes ternary drug-gene-mutation interactions and binary drug-mutation interactions, respectively. “S” and “C” indicates accuracy reported on instances within single sentences and considering all instances, respectively. K in GCN indicates that pre-processed pruned trees include tokens up to distance K away from the dependency path in the LCA subtree. * indicates results from previous work (Guo et al., 2019)

Model	**F1-score**	**Model**	**F1-score**
SVM (Rink and Harabagiu, 2010)	82.2	PA-LSTM (Zhang et al., 2017)	82.7
SDP-LSTM (Xu et al., 2015)	83.7	C-GCN (Zhang et al., 2018)	84.8
SPTree (Miwa and Bansal, 2016)	84.4	AGGCN (Guo et al., 2019)	85.7
C-GCN-MG (our model)	**85.9**		

Table 2: Results on SemEval dataset

2019); and (b) sequence-based models: which consider SoTA Position Aware LSTM (PA-LSTM) (Zhang et al., 2018) model. As shown in Table 2, C-GCN-MG achieves higher performance (F1-score of 85.9) against all baselines on the SemEval dataset. However, on TACRED dataset (Table 3), the performance of C-GCN-MG is comparable to the baselines. In particular, C-GCN-MG outperforms sequence-based PA-LSTM, GCN and AGGCN models. However, C-GCN-MG achieves a lower performance in comparison to AGGCN on TACRED dataset.

6.3 Single graph vs. Multiple sub-graphs

To further evaluate the contribution of multiple graphs for relation extraction, we examine the performance of the following two models: (a) C-GCN-SG: A C-GCN using a single graph (SG) constructed using the nodes in the SDP between the entities; and (b) C-GCN-MG: A C-GCN using multiple graphs (MG) constructed using (a) nodes in SDP; and (b) nodes associated with the entity mentions in the dependency graph. Table 4 shows that C-GCN-MG significantly outperforms C-GCN-SG, both in terms of precision and recall, by scoring an F1-score of 85.91 against 83.73 ($p \leq 0.05$ under the Wilcoxon Signed-Rank Test). Using multiple graphs facilitates inclusion of additional nodes that help in identifying relations more accurately, while providing better coverage by retrieving more relevant instances. These results further confirms our hypothesis that using multiple sub-graphs with GCNs are more useful than using a single graph.

6.4 Contribution of multiple sub-graphs

To further study the contribution of multiple sub-graphs (C-GCN-MG) against single graph (C-GCN-SG), sentences in SemEval test set are categorised into three groups using distance between entities. The average number of tokens (μ) and the standard deviation (σ) over different lengths of tokens between e_1 and e_2 are used to obtain three different groups of sentences: (a) *short-distance spans* ($j \leq \mu - \sigma$), (b) *medium-distance spans* ($\mu - \sigma < j < \mu + \sigma$), and (c) *long-distance spans* ($j \geq \mu + \sigma$), where j

Model	**P**	**R**	**F**	**Model**	**P**	**R**	**F**
LR (Zhang et al., 2017)	73.5	49.9	59.4	GCN (Zhang et al., 2018)	69.8	59.0	64.0
SDP-LSTM (Xu et al., 2015)	66.3	52.7	58.7	C-GCN (Zhang et al., 2018)	69.9	63.3	66.4
Tree-LSTM (Tai et al., 2015)	66.0	59.2	62.4	AGGCN (Guo et al., 2019)	69.9	60.9	65.1
PA-LSTM (Zhang et al., 2018)	65.7	64.5	65.1	C-AGGCN (Guo et al., 2019)	71.8	66.4	**69.0**
C-GCN-MG (our model)	68.0	64.4	66.1				

Table 3: Results on TACRED dataset. P: Precision, R: Recall, F: F1-score

Model	Precision %	Recall %	F1-Score %	Model	Precision %	Recall %	F1-Score %
C-GCN-SG	83.01	84.64	83.73	C-GCN-MG	84.43	87.56	**85.91***

Table 4: Performance of models using single graph and multiple sub-graphs on SemEval test set.

is the number of tokens between e_1 and e_2. The average number of tokens μ was found to be 3 and the standard deviation σ obtained was 9. The numbers of sentences in different categories shown in Table 5, shows that a large proportion of sentences (ca. 72%) are categorised as MEDIUM SPANS with sentences having $6-12$ tokens between entities. The performance of C-GCN-MG and C-GCN-SG across different spans of sentences in SemEval test set is provided in Table 6, shows that C-GCN-MG achieves higher performance consistently across all three spans of sentences over C-GCN-SG. These results shows that multiple sub-graphs are extremely useful, particularly for LONG SPAN sentences with large distances between the entities. The contribution of C-GCN-MG is also significant for sentences in the MEDIUM SPANS category. However, for SHORT SPANS sentences the difference in the performance of C-GCN-MG and C-GCN-SG is less significant, indicating that a single graph is sufficient for shorter sentences.

6.5 Number of nodes used in the graph

The C-GCN-MG and C-GCN-SG models are evaluated using path-centric pruning distance (Zhang et al., 2018) in the following three settings to examine the effect of expanding nodes in SDP: (a) $K = 0$: using entities in SDP; (b) $K = 1$: include nodes directly connected to nodes in SDP; and (c) $K = \infty$: use full dependency graph. The evaluation results provided in Table 7 shows that as an overall trend, the F-scores of both the models drops significantly when the nodes in the graph are added incrementally by expanding the SDP. Both C-GCN-MG and C-GCN-SG models achieve the best F-scores by using only the nodes in the SDP, corresponding to C-GCN-MG ($K = 0$) and C-GCN-SG ($K = 0$) in Table 7. The superior performance by scoring a higher recall without compromising on precision, indicates the usefulness of limiting to a minimum set of nodes in the graph. Although the use of higher number of nodes facilitates achieving higher precision, the models largely suffer from poor recall resulting in lower F-scores. Interestingly, considering multiple sub-graphs as done by C-GCN-MG models leads to better F-scores in comparison to the corresponding C-GCN-SG models, confirming our hypothesis to use multiple sub-graphs for relation extraction. The visualisation of the attention weights of nodes at the final attention layer by C-GCN-MG for a an example sentence is shown Figure 4, indicates that the model restricting to nodes in the SDP (4b.1) with lesser number of nodes facilitates finer representations for each word in the graph, with nodes besides entity mentions also contributing to the task. In the associated sub-graphs, higher weights are assigned to e_1 (school), whereas e_2 receives a lower weight, indicating that the model uses information from the corresponding sub-graphs. In contrast, when a model uses a larger set of nodes (Figure 4c.1), the contribution of words in the graph is less evident.

6.6 Ablation study

The ablation study examining different feature sets is provided in Table 8. As seen, BERT helps to achieve a significant improvement in the overall performance. The syntactic embeddings (POS, NER, DEP, word-type (WT), edge weights) helps to increase performance. In addition to syntactic embeddings, the use of edge weights are observed to boost the F1-score to 85.9. The contribution of the contextual BiLSTM layer is also significant in the overall performance, without which the model achieves a low F1-score of 78.2, showing the importance of using contextual layer to derive node features for GCN.

SHORT SPANS	MEDIUM SPANS	LONG SPANS	TOTAL
$(j \le \mu - \sigma)$	$(\mu - \sigma < j < \mu + \sigma)$	$(j \ge \mu + \sigma)$	
365	1966	386	2717
13.50 (%)	72.30 (%)	14.20 (%)	

Table 5: Details of different categories of short, long and medium spans in the SemEval test set.

Model	Precision %	Recall %	F1-Score %	Model	Precision %	Recall %	F1-Score %	Model	Precision %	Recall %	F1-Score %
	SHORT SPANS				MEDIUM SPANS				LONG SPANS		
C-GCN-SG	79.66	78.18	78.65	C-GCN-SG	83.81	85.88	84.73	C-GCN-SG	76.48	80.59	77.94
C-GCN-MG	80.63	81.01	**79.95**	C-GCN-MG	85.16	88.46	**86.72**	C-GCN-MG	81.78	85.95	**83.50***

Table 6: Performance of models across short, medium and long spans in SemEval test set.

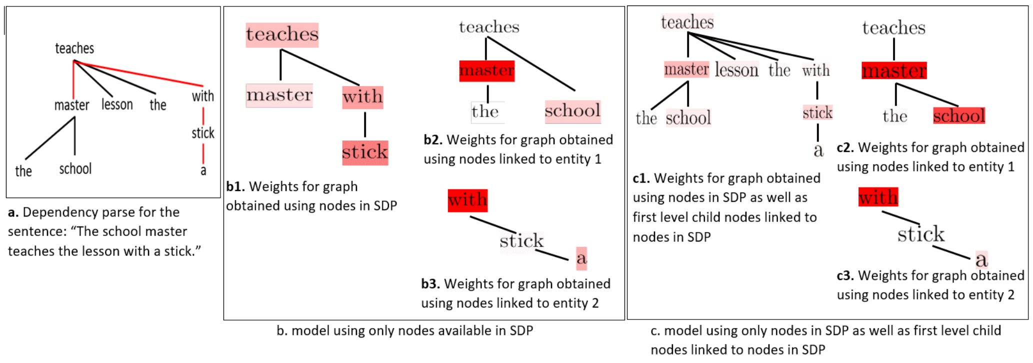
teaches
master
lesson
the
with
stick
the
school
a
a. Dependency parse for the sentence: "The school master teaches the lesson with a stick."
teaches
master
with
stick
b1. Weights for graph obtained using nodes in SDP
teaches
master
the
school
b2. Weights for graph obtained using nodes linked to entity 1
with
stick
a
b3. Weights for graph obtained using nodes linked to entity 2
b. model using only nodes available in SDP
teaches
master
lesson
the
with
the
school
stick
a
c1. Weights for graph obtained using nodes in SDP as well as first level child nodes linked to nodes in SDP
teaches
master
the
school
c2. Weights for graph obtained using nodes linked to entity 1
with
stick
a
c3. Weights for graph obtained using nodes linked to entity 2
c. model using only nodes in SDP as well as first level child nodes linked to nodes in SDP

Figure 4: Visualisation of attention weights learnt by C-GCN-MG model using different sets of nodes. Words are shaded based on contribution of weights obtained at the final attention layer.

Model	Precision %	Recall %	F1-Score %	Model	Precision %	Recall %	F1-Score %
		C-GCN-SG				C-GCN-MG	
C-GCN-SG ($K = 0$)	83.01	84.64	**83.73**	C-GCN-MG ($K = 0$)	84.43	87.56	**85.91***
C-GCN-SG ($K = 1$)	80.38	82.13	81.22	C-GCN-MG ($K = 1$)	86.74	84.05	85.21
C-GCN-SG ($K = \infty$)	82.91	77.99	80.16	C-GCN-MG ($K = \infty$)	86.83	82.41	84.43

Table 7: Performance of models using different set of nodes in the graph.

Model	F1
BEST C-GCN-MG	85.9
– Attention (use 3−GCN+Pooling layer)	85.7
– POS, NER, DEP, WT	82.7
– BERT (use $300d$-GloVe (Pennington et al., 2014) embeddings)	82.4
– Edge Weights	85.2
– BiLSTM	78.2

Table 8: Ablation study of the best C-GCN-MG on the SemEval test set.

7 Conclusion

To conclude, we proposed a contextualised GCN model using multiple sub-graphs representing a sentence as against a single graph to improve relation extraction. The proposed model scores both in terms of precision and recall to achieve a higher accuracy and better coverage. The improvement in performance is observed to largely come from accurately predicting relations for instances with large distance between entities. The superior performance of the model against SoTA graph-based models on standard relation extraction datasets clearly establish the strength of the proposed model.

References

Lars Backstrom and Jure Leskovec. 2011. Supervised random walks: predicting and recommending links in social networks. In *Proc. of WSDM*, pages 635–644. ACM.

Jie Bao, Tianfu He, Sijie Ruan, Yanhua Li, and Yu Zheng. 2017. Planning bike lanes based on sharing-bikes' trajectories. In *Proc. of KDDM*, pages 1377–1386. ACM.

Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. 2005. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, pages 2224–2232.

Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs*.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2019. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *arXiv preprint arXiv:1911.10422*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

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

Angrosh Mandya, Danushka Bollegala, Frans Coenen, and Katie Atkinson. 2018. Combining long short term memory and convolutional neural network for cross-sentence n-ary relation extraction. *arXiv preprint arXiv:1811.00845*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *TACL*, 5:101–115.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*, pages 1532–1543.

Chris Quirk and Hoifung Poon. 2016. Distant supervision for relation extraction beyond the sentence boundary. *arXiv preprint arXiv:1609.04873*.

Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 256–259. Association for Computational Linguistics.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proc. of EMNLP*, pages 1201–1211. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. N-ary relation extraction using graph state lstm. *arXiv preprint arXiv:1808.09101*.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *Proc. of EMNLP*, pages 1785–1794.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proc. of EMNLP*, pages 35–45.

Yuhao Zhang, Peng Qi, and Christopher D Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*.