Context Based Entity Graph Convolutional Network for Multi-hop Reading Comprehension

Lunhua Zhang[a*], Xiaohong Liu[b]

Beijing University of Posts and Telecommunications

[a]lunhuazhang@bupt.edu.cn, [b]xiaohongliu@bupt.edu.cn

Keywords: Multi-hop reading comprehension; Graph convolutional network; Memory network; Multi-granularity encodings

Abstract: Multi-hop reading comprehension poses new challenge on machine reading comprehension tasks which requires reasoning between multiple documents. In this paper, we propose a graph-based model called Context based Entity Graph Convolutional Network (CEG). In order to take full advantage of context information, on the one hand we apply multi-granularity encodings for entity which capture rich context information, on the other hand we extract surrounding context of entity to enrich entity encoding. The surrounding context of entity will further be utilized by Memory Network to support graph reasoning. Experimental evaluation shows CEG achieves competitive performance on the QAngaroo WIKIHOP dataset, and the following ablation test demonstrates our proposed context extraction modules are effective in multi-hop reading comprehension.

1. Introduction

Machine Reading Comprehension (MRC) requires the model to answer a natural language question given relevant context. The advent of large-scale datasets, such as SQuAD[1] and CNN/Daily mail[2], greatly promotes the development of the field. Many end-to-end deep neural models such as BiDAF[3] and SAN[4] are constructed to tackle the problem, achieving great success especially after the release of the BERT[5]. But the challenge still remains, many of these datasets are specified to single-hop reading comprehension, in which the answer can be concluded by only reading a single paragraph or sentence. While in many real scenarios, people have to reason across multiple documents in order to find the answer.

The WIKIHOP[6] dataset was proposed exactly to meet the above challenge. Each sample in WIKIHOP contains multiple supporting documents, and the goal is to select the correct answer in a set of candidates for a query. The dataset is carefully constructed to ensure that most queries cannot be answered by only reading a single document. Multi-step reasoning chains across documents are needed to find the correct answer which poses great challenge for previous models. Some baseline models such as BiDAF[3] and DCN[7], which have fine performance in single-hop MRC tasks, suffer dramatic accuracy decline in this task.

In this paper, we propose a multi-hop MRC model, named Context based Entity Graph Convolutional Network (CEG). Based on graph convolutional network (GCN), the model performs graph reasoning to learn relation-aware representation of nodes. There are two types of nodes in graph, entity nodes and candidate nodes. Entity nodes are mentions of candidate nodes which are extracted from supporting documents. Attention mechanism is utilized to attend query information into nodes. To encode rich context information, multi-granularity encodings are applied. Furthermore, entity nodes are coupled with supporting context which will be utilized by Memory Network to support graph reasoning.

Experimental results demonstrate that CEG achieves competitive performance on the WIKIHOP dataset. Ablation test also shows CEG benefits from rich contextual encoding, attention mechanism and graph convolutional network.

Our contributions in this work are as follows:

Utilizing multi-granularity encodings to acquire context information at different level.

Extracting surrounding context of entity nodes to make full use of context information in supporting documents.

Applying Memory Network to extract relevant context information of nodes to support graph reasoning.

2. Related Work

Recently, graph-based models are proved to be efficient in the task of multi-hop MRC. MHQA-GRN[8] and Entity-GCN[9] are the first attempts in this direction. They extract entities from documents to make entity graph. Then multi-hop reasoning is processed between document nodes to learn complex relationship among entities so as to derive relational information for answering the query. BAG[10] introduces a new bi-directional attention which encodes mutual information between graph nodes and query. CFC[11] uses co-attention and self-attention to do coarse and fine reasoning in documents without using graph. HDE[12] proposes heterogeneous document-entity graph which contains different types of query-aware nodes representing different granularity levels of information.

The models above can be divided into two categories. One utilizes only nodes information using pretrained contextual embedding such as ELMo[13]. Entity-GCN and BAG fall into this category. The other makes use of full document context followed by attention to extract relevant information. These models tend to use simple word embedding such as GLoVe[14] considering training cost. CFC and HDE are representative models in this category. Our model differs from those models in that it stands in the middle of the two categories and combines the advantages of them. On the one hand, the pretrained BERT embedding is applied due to the superior performance in contextual encoding. Note that BERT embedding is limited to query, entities and candidates due to memory limits. The BERT embedding is generated in offline step which does not effect the efficiency of online training. On the other hand, we extract nodes surrounding tokens as supporting context. The supporting context will be used by memory network to enrich node encoding.

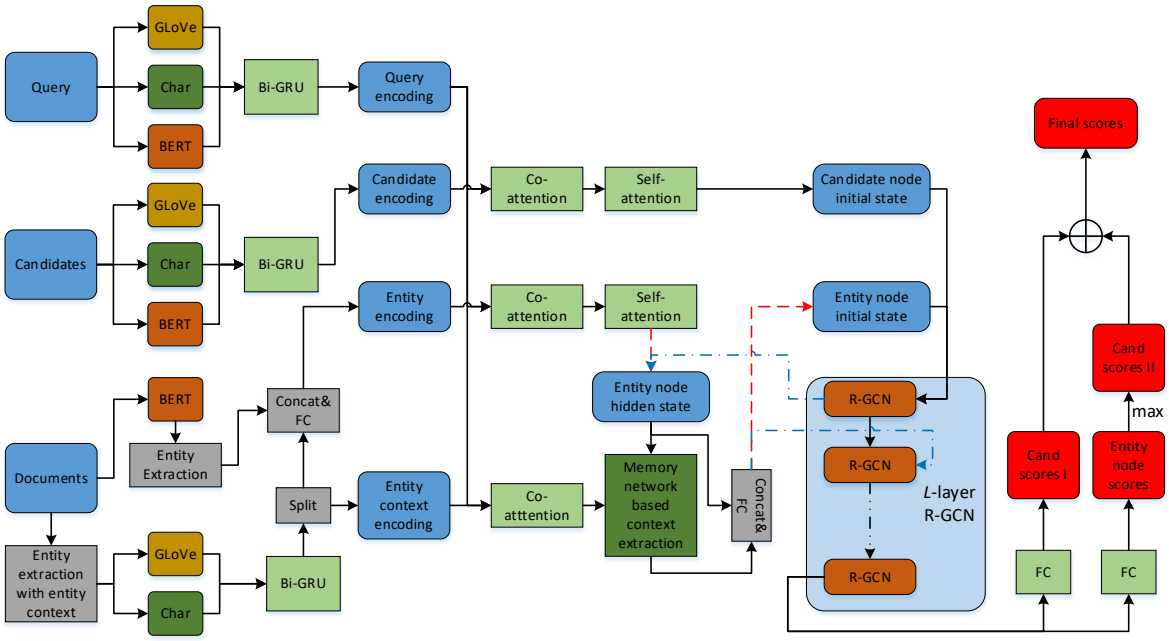

Fig. 1. Framework of CEG model. Red dash lines apply for first layer in GCN, blue dash dot lines apply for other layers.

3. Model

We first introduce the task of WIKIHOP[6] dataset. A sample comprises of a query, a set of supporting documents and a set of candidate answers. The query is in form of (s, r, ?) which

represents subject, relation and unknown object respectively. The dataset is created to guarantee the unknown query object and candidate answers are entities in supporting documents. In fact, there is at least one reasoning chain which starts from the query subject and runs through several candidates between documents to arrive at the final answer. The goal of the model is to find the right answer in candidates through document reasoning.

The proposed CEG model are shown in Fig. 1. It can be categorized into four parts: (A) contextual encoding, (B) graph construction and reasoning, (C) memory network based contextual extraction, (D) output layer.

3.1. Contextual encoding

Entity extraction. In multi-hop reading comprehension scenarios, multiple documents are required to answer a query. Simply concatenating the documents and applying a previous neural MRC model does not achieve promising performance yet with enormous resource costs. Graph models are based on entity nodes which do not require the full document context. As mentioned above, query answer is in candidate set and candidates are entities in documents. A simple string matching heuristic is used to find candidate mentioned in documents as entity nodes. Now that we have the start and end position of every entity node in its corresponding supporting document, surrounding tokens around the entity node mention are extracted as supporting context for the entity node. Finally, a set of entity nodes are obtained. For entity node i, we define the node content as e_i, the node supporting context as s_i, together they make up the whole context w_i. The length of the whole context w_i is fixed which defined as l_w and the length of surrounding tokens on both sides should keep consistent if condition allows, so that context information from forward direction and backward direction can both be acquired.

Multi-granularity context encoding. MRC tasks require the model to gather context information to have a thorough understanding of documents and query, so we apply multi-granularity encoding trying to capture context information at different level. First, the pretrained word embedding GLoVe[14] is used to represent tokens at word level. Then we apply character ngram vectors[15] to encode tokens at character level which also offset the low word coverage of GLoVe. Moreover, a strong pretrained model BERT[5] is utilized to acquire rich contextual encoding.

For query q, glove, char and bert embedding matrix are concatenated as $X_q \in \mathbb{R}^{l_q \times d_q}$ where l_q is the length of query tokens and $d_q = d_{glove} + d_{char} + d_{bert}$. Then bidirectional recurrent neural network (RNN) with gated recurrent unit (GRU) is applied to encode the contextual information in query. The output is $H_q \in \mathbb{R}^{l_q \times h}$, where h denotes the output dimension of RNN encoder. For candidate node c, similar method is applied to get $H_c \in \mathbb{R}^{l_c \times h}$ where l_c is the length of candidate node tokens. For entity node e and entity node supporting context s, glove and char embedding matrix of the whole context w are concatenated as $X_w \in \mathbb{R}^{l_w \times d_w}$ where l_w is the length of whole context and $d_w = d_{glove} + d_{char}$. We also use BiGRU to encode the whole context to get $H_w \in \mathbb{R}^{l_w \times h}$, then split it into $T_e \in \mathbb{R}^{l_e \times h}$ and $H_s \in \mathbb{R}^{l_s \times h}$. l_e and l_s represent the length of entity node tokens and the length of entity node supporting context tokens respectively, and $l_e + l_s = l_w$. It should be noted that BERT embedding for entity node $B_e \in \mathbb{R}^{l_e \times d_{bert}}$ is calculated based on original documents, then extracted according to the position indices of the node tokens. Due to resource limits, BERT embedding for entity nodes supporting context is not used. B_e and T_e are concatenated column-wise and transformed into $H_e \in \mathbb{R}^{l_e \times h}$ via a 1-layer linear network. At last, we get contextual encoding H_q, H_e, H_s, H_c for query, entity node, entity node supporting context and candidate node respectively.

Co-attention. Co-attention is widely used in single-hop reading comprehension tasks. Recently it was applied to multi-hop reading comprehension. It is responsible for generating mutual information between query and entity nodes (entity nodes supporting context or candidate nodes). Take a query and a entity node for illustration. Co-attention combines learned query contextual information attended by entity node and entity node contextual information attended by query.

Given RNN encoded sequences of a query $H_q \in \mathbb{R}^{l_q \times h}$ and a entity node $H_e \in \mathbb{R}^{l_e \times h}$. The

similarity matrix between query and entity node is calculated via

$$A_{qe} = H_e(H_q)^T \in \mathbb{R}^{l_e \times l_q} \quad (1)$$

Then the attended entity node and query can be derived by

$$S_e = \mathrm{softmax}(A_{qe})H_q \in \mathbb{R}^{l_e \times h} \quad (2)$$

$$S_q = \mathrm{softmax}(A_{qe}^T)H_n \in \mathbb{R}^{l_q \times h} \quad (3)$$

where $\mathrm{softmax}(\cdot)$ represents column-wise normalization. Next the attended query is further co-attended followed by a linear projection:

$$R_e = \mathrm{MLP}(\mathrm{softmax}(A_{qe})S_q) \in \mathbb{R}^{l_e \times h} \quad (4)$$

$\mathrm{MLP}(\cdot)$ is a one-layer MLP with tanh activation. The final co-attention context can be obtained via

$$Z_e = \mathrm{MLP}(\mathrm{concat}(H_e, S_e, H_e \circ S_e, H_e \circ R_e)) \in \mathbb{R}^{l_e \times h} \quad (5)$$

where $\mathrm{MLP}(\cdot)$ is a one-layer MLP with tanh activation. $\mathrm{concat}(\cdot)$ stands for column-wise concatenation. The operator $\circ$ is element-wise multiplication.

The co-attended context Z_e is expected to carry query-aware contextual information of the entity node. Similar method can be applied to query and entity nodes supporting context, and query and candidate nodes to get Z_s and Z_c.

Self-attention. Self-attention performs context summarization by calculating a normalized score for every token in the context and compute a weighted sum over the context encoding. In entity graph, node should be a fixed-length non-sequential vector while co-attended context is sequential. So self-attention is applied to summarize the co-attention output of entity nodes and candidate nodes. Formally, given Z_e as input, the self-attention can be formulated as

$$a_e = \mathrm{softmax}(\mathrm{MLP}(Z_e)) \in \mathbb{R}^{l_e \times 1} \quad (6)$$

$$z_e = Z_e^T a_e \in \mathbb{R}^h \quad (7)$$

where $\mathrm{MLP}(\cdot)$ is a two-layer MLPs with tanh activation. The hidden layer size is half of the input dimension, and the output dimension is 1. The output z_e provides a summary of co-attended sequential vector Z_e. Similarly, we can get candidate node summary z_c using self-attention.

3.2. Graph construction and reasoning

Graph construction. Graph is made up of nodes and edges. The graph is undirected so that the information can be propagated in both directions. There are two types of nodes in graph: entity nodes and candidate nodes. Entity nodes are mentions of candidates from documents. Inspired by HDE[12], we add candidate nodes to gather and summarize information from different entity nodes of the same mention. For edge connections, different types of edges are defined to encode various structural information in the graph:

1) Entity node pair of different mentions are connected if they are extracted from the same document.

2) Entity node pair of the same mention are connected if they are extracted from different documents.

3) An entity node and a candidate node are connected if the entity node is a mention of the candidate node.

4) All candidate nodes connect with each other.

5) Entity nodes that do not meet the previous conditions are connected.

Message passing. Relational Graph Convolutional Network (R-GCN)[16] is applied to propagate message across graph nodes and generates relational representations of nodes. R-GCN has multiple layers which correspond to multi-hop graph reasoning. Nodes can aggregate message passed from neighboring nodes, and a message is specific to a certain edge type (also referred as relation). At first

layer, we initialize the self-attention output as node representation o_i^1. Then, at lth layer, we use memory network to extract relevant context information (described in next section) for entity node, which will be concatenated with raw entity node representation to form new node representation. Candidate node representation remains unchanged:

$$g_i^l = \begin{cases} f_s\left(\text{concat}\left(o_i^l, MN(Z_s, o_i^l)\right)\right) & \text{i is entity node} \\ o_i^l & \text{i is candidate node} \end{cases} \quad (8)$$

$g_i^l \in \mathbb{R}^h$ is new node representation for node i at lth layer. $MN(\cdot)$ stands for memory network which will be introduced in next section. Z_s is supporting context for entity node i which is generated in co-attention and keeps constant during graph reasoning. f_s represents linear transformation to keep node dimension constant. h is node vector dimension.

After obtaining node representation, R-GCN performs aggregation to gather information from neighbors of each node, then combines node information to generate update information. The process can be formulated as

$$u_i^l = W_0^l g_i^l + \sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{N}_i^r|} \sum_{j \in \mathcal{N}_i^r} W_r^l g_j^l \quad (9)$$

where $\mathcal{R}$ is the set of all edge types and $\mathcal{N}_i^r$ is the set of indices of the neighboring nodes that share type r edge connection with node i. The symbol $|\cdot|$ denotes size of a set. $W_r^l \in \mathbb{R}^{d \times d}$ is a weight matrix specific to edge type. $W_0^l \in \mathbb{R}^{d \times d}$ applies linear transformation the node representation.

To avoid smoothing problem in GNN[17], a gating mechanism is applied to control the amount of update information that should propagate to the next hop:

$$p_i^l = \sigma(f_g([g_i^l, u_i^l])) \quad (10)$$

where $\sigma(\cdot)$ is sigmoid function and f_g is linear transformation. The symbol $[\cdot,\cdot]$ denotes column-wise concatenation. Finally, the node representation in the next layer is defined as gated combination of a non-linear transformed update message and the previous representation:

$$o_i^{l+1} = p_i^l \circ \tanh\left(u_i^l\right) + \left(1 - p_i^l\right) \circ g_i^l \quad (11)$$

where $\circ$ denotes element-wise multiplication. It should be noted that all parameters are shared layer to layer in order to decrease model complexity.

3.3. Memory network based contextual extraction

Memory network stores knowledge source in memory, and uses a state vector to extract relevant information from source to support further reasoning. Miller et al.[18] apply Key-Value Memory Network to perform QA. Inspired by their work, we apply memory network to extract contextual information that cannot be directly encoded in nodes representation.

Knowledge source stored in memory is the entity node supporting context Z_s, state vector is the raw entity node representation o. Note that in every step of graph reasoning, memory network module will be performed for every entity node simultaneously, so we ignore superscript l and subscript i in o_i^l which indicates the representation of entity node i in lth layer.

Key addressing. Key and value are both Z_s in our memory network. During key addressing, each word in the supporting context is assigned a relevance probability by comparing each word to the state vector:

$$p = \text{softmax}(Z_s o) \quad (12)$$

Value reading. Value reading result is the weighted sum of memory value using key addressing probability:

$$m = Z_s^T p \quad (13)$$

The whole memory network module can be denoted as

□ □ $\mathrm{MN}(Z_s, o) \rightarrow m$ □ □ □ □ (14)

3.4. Output layer

After graph reasoning, final representation of entity nodes and candidate nodes are obtained. Note that entity node is mention of corresponding candidate node in original document. The candidate score is calculated via

□ $p = \mathrm{softmax}(f_C(O^C) + \mathrm{cmax}(f_E(O^E)))$□ □ □ (15)

where $O^C \in \mathbb{R}^{N_C \times h}$ is the final representation of candidate nodes and N_C is the number of candidates. $O^E \in \mathbb{R}^{N_E \times h}$ is the final representation of entity nodes and N_E is the number of entity nodes. $\mathrm{cmax}(\cdot)$ is defined to take the max score over entity nodes that belong to the same candidate. f_C and f_E are two-layer MLPs with tanh activation. The hidden layer size is half of the input dimension, and the output dimension is 1. Candidate node score and entity node score are summed as final candidate score. Softmax is then applied to generate the probability distribution which indicates the probability of each candidate becoming the answer. The loss function is defined as the cross entropy between one-hot answer vector and the predicted probability.

4. Experiment

We use only the unmasked version of the WIKIHOP dataset. There are 43738 samples in the training set, 5129 samples in the development set and 2451 samples in the test set. The test set is not public so the model will be evaluated blindly.

In the experiment, we use base-cased BERT pretrained model and generate 768-dimensional embedding for query, entities and candidates offline. Besides, 300-dimensional GLoVe embedding and 100-dimensional character n-gram embedding are used to encode entity supporting context, query and candidates. The encoding size h is 200. The length of whole context is fixed as 64. The number of GCN layer is set as 5. In addition, dropout with rate 0.2 is applied before attention and before graph reasoning. We use Adam as optimizer. The learning rate is initialized as 2×10^{-4} which will be halved for every 5 epochs. We train the model for 20 epochs.

Table 1 The performance of different models on WIKIHOP dataset

Model	**Accuracy(%)**	
	Dev	**Test**
BiDAF	-	42.9
Entity-GCN	64.8	67.6
CFC	66.4	70.6
BAG	66.5	69.0
HDE	68.1	70.9
Our Model	69.6	73.6

As shown in Table 1, the proposed CEG model achieves competitive performance compared to the previously published models. It is much better than the baseline BiDAF[7] model presented in the WIKIHOP paper. Compared to the previous graph-based models such as Entity-GCN[10], BAG[11] and HDE[13], CEG surpasses them by a large margin. Note that the accuracy improvement in test set is more substantial than in development set, proving the robustness of the model.

Table 2 Ablation results on WIKIHOP dev set

Model	**Accuracy (%)**	
	Dev	**Δ**
Full Model	69.6	
- BERT	66.8	2.8
- GLoVe & Char	68.6	1.0

Model	Accuracy (%)	
	Dev	Δ
- Entity node context	67.1	2.5
- Candidate nodes	68.4	1.2
- Attention	66.9	2.7
- GCN	64.7	4.9

Ablation results are shown in Table 2. First, we verify the effectiveness of two context encoding methods we propose. Without BERT embedding and GLoVe&Char embedding, the accuracy on WIKIHOP development set drops 2.8% and 1.0% respectively. This indicates the efficiency of multi-granularity encoding. If we do not use entity node context, the accuracy shows 2.5% absolute drop. The results above prove the contextual information is vital for reasoning.

Then, we investigate the effect of other modules in CEG model. Once the candidate nodes are removed, the accuracy has 1.2% drop. We suspect it is because candidate nodes can gather information from the entity nodes of the same mention, it performs as a sort of summarization which can improve model performance. If the attention is replaced by concatenation of query and nodes (nodes context), the model performance is dropped by 2.7%, which reflects the importance of attention mechanism. If GCN is removed and the output of self-attention is directly used in output layer, the accuracy drops 4.9%. It is the largest drop in ablation test, which indicates the GCN plays fundamental roles in the model.

5. Conclusion

A graph-based model CEG is proposed for multi-hop reading comprehension which focuses on making more use of context information from documents. To achieve the goal, two main methods are applied. One is using multi-granularity embedding to acquire encoding at different level, and the strong contextual embedding BERT is utilized in this model. The other is to extract supporting context for entity nodes in documents which is used as source for memory network to generate context information to support reasoning. The experimental results demonstrate the effectiveness of the two proposed methods, and the model achieves competitive performance in WIKIHOP dataset. In the future, we would like to further investigate more powerful ways to encode the context information into graph nodes, and try to apply the methods to other tasks in MRC.

Acknowledgements

This work is supported by NSFC 61773037. Special thanks also give to Johannes Welbl for helping me evaluating the model.

References

[1] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.

[2] Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. In Advances in neural information processing systems (pp. 1693-1701).

[3] Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603.

[4] Liu, X., Shen, Y., Duh, K., & Gao, J. (2017). Stochastic answer networks for machine reading comprehension. arXiv preprint arXiv:1712.03556.

[5] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[6] Welbl, J., Stenetorp, P., & Riedel, S. (2018). Constructing datasets for multi-hop reading comprehension across documents. Transactions of the Association for Computational Linguistics, 6, 287-302.

[7] Xiong, C., Zhong, V., & Socher, R. (2016). Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604.

[8] Song, L., Wang, Z., Yu, M., Zhang, Y., Florian, R., & Gildea, D. (2018). Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. arXiv preprint arXiv:1809.02040.

[9] De Cao, N., Aziz, W., & Titov, I. (2018). Question answering by reasoning across documents with graph convolutional networks. arXiv preprint arXiv:1808.09920.

[10] Cao, Y., Fang, M., & Tao, D. (2019). BAG: Bi-directional Attention Entity Graph Convolutional Network for Multi-hop Reasoning Question Answering. arXiv preprint arXiv:1904.04969.

[11] Zhong, V., Xiong, C., Keskar, N. S., & Socher, R. (2019). Coarse-grain fine-grain coattention network for multi-evidence question answering. arXiv preprint arXiv:1901.00603.

[12] Tu, M., Wang, G., Huang, J., Tang, Y., He, X., & Zhou, B. (2019). Multi-hop Reading Comprehension across Multiple Documents by Reasoning over Heterogeneous Graphs. arXiv preprint arXiv:1905.07374.

[13] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. arXiv preprint arXiv:1802.05365.

[14] Pennington, J., Socher, R., & Manning, C. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

[15] Hashimoto, K., Xiong, C., Tsuruoka, Y., & Socher, R. (2016). A joint many-task model: Growing a neural network for multiple nlp tasks. arXiv preprint arXiv:1611.01587.

[16] Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., & Welling, M. (2018, June). Modeling relational data with graph convolutional networks. In European Semantic Web Conference (pp. 593-607). Springer, Cham.

[17] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[18] Miller, A., Fisch, A., Dodge, J., Karimi, A. H., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. arXiv preprint arXiv:1606.03126.