

Graph To Coherent Text: Passage Generation from Knowledge Graphs by Exploiting Edge Representations in Sentential Contexts

Abstract

In the present digital age, much information is organized as knowledge graphs, where numerous techniques have been developed to retrieve relevant subgraphs. As the information represented in such retrieved subgraphs is not readily suitable for human laymen users to understand, it is useful to develop approaches to represent such information in natural language. In this work, we propose a neural modeling framework that jointly learns to (1) generate topically coherent and informative text by computing the representation of the input knowledge graph for each sentential context, and to (2) generate text in a sentence-by-sentence order to improve tractability for long sequence generation. Our proposed approach improves the accuracy of text generation using objective metrics such as BLEU and METEOR in comparison to established baselines such as graph transformer networks on the AGENDA dataset. Human evaluations also show that our proposed model compares favorably with established baselines on the dimensions of informativeness and coherence.

Introduction

Knowledge graphs are a specific form of graphs in which entities are represented as nodes while the relationships between them are represented as labeled edges (see Figure 1 for an example). Knowledge graphs are a useful way to organize comprehensive information consisting of a large quantity of entities and relations between them. Knowledge graphs have been a successful means of managing semantic information (Wang et al. 2018). Research in developing methods to construct knowledge graphs have been vibrant (e.g. (Mehta, Singhal, and Karlapalem 2019).) Much efforts are also made to query large knowledge graphs for retrieving relevant information (Bao et al. 2016), where often the retrieved results are subgraphs of the knowledge graph (Kasneci, Elbassuoni, and Weikum 2009). In such scenarios, approaches to represent such subgraphs in natural language to make the retrieved results human readable is highly desirable in developing user-friendly interactive technology. Figure 1 illustrates an example of such a task, where the inputs are a knowledge graph and a proposed title, and the corresponding generated output text.

Natural language generation approaches to convert knowledge graphs to human-readable textual descriptions is spurring a lot of research interest. Recent efforts include (Marcheggiani and Perez-Beltrachini 2018), where the majority of the work is intended for generating single sentences from the input graph. A lucid generated piece of text should be coherent and informative while being grammatically accurate. Technical challenges cascade when one attempts to generate long sequences (e.g. multiple sentences) of text, as information coverage, fluency, and coherence all become more challenging to model properly. Koncel-Kedziorski et al. (2019) is the first to generate multiple-sentence text from knowledge graphs accompanied with a dataset release. They used a graph transformer network to process the information in graphs to human readable text.

In our work, we are motivated by computational linguistic approaches of modeling coherence when considering multi-sentences (Barzilay and Lapata 2008). In particular, we exploit topic continuity for modeling textual coherence as inspired by (Pu 2006). Our proposed neural architecture generates coherent multi-sentences from knowledge graphs (Koncel-Kedziorski et al. 2019) by (1) explicitly modeling content order and decoding sentence-by-sentence (as opposed to a single sequence of multiple-sentences, which is the default in neural generation frameworks) to better model the order of the content and make the generated text more tractable, reducing the probability of brittle yet common neural language generation behavior such as repeating text, contradictory information and bland, non-specific descriptions¹, and by (2) explicitly learning an optimum trade off between topical continuity for coherence and novel content selection for informativeness during generation process.

Related Work

Knowledge graph construction has played an important role in organizing and managing semantic information, where much recent research focuses on end-to-end deep learning techniques (Mehta, Singhal, and Karlapalem 2019). Data modeling approaches using such structured information sources to support more intelligent systems and enable user-friendly interactions has also been a recent research trend. For example, some researchers have used knowledge

¹<https://newgeneralization.github.io/slides/YejinChoi.pdf>

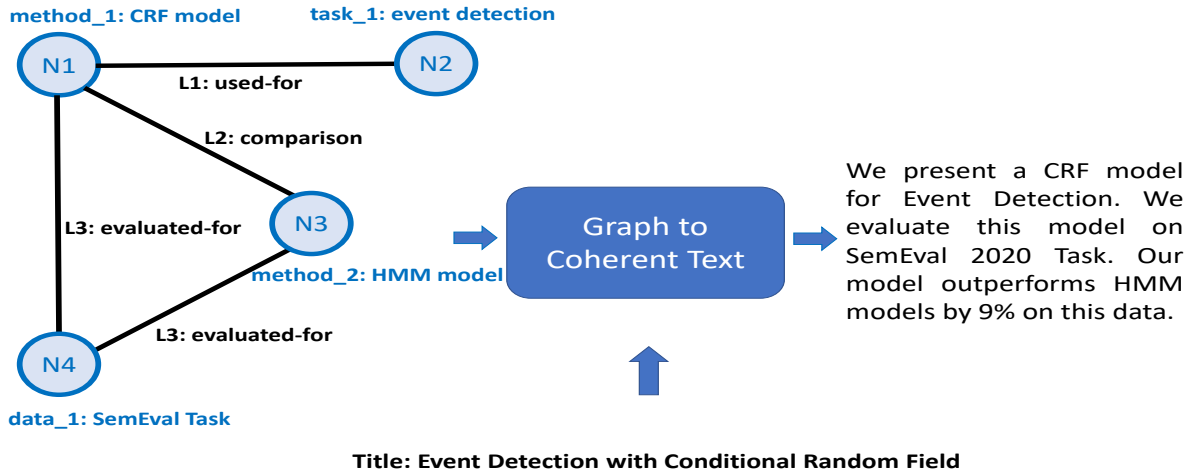


Figure 1: An example of natural language description generated from a knowledge graph guided by the text of the title.

graphs to improve question answering techniques (Zhang et al. 2018), while others have worked on methods to retrieve answers in a presentable format (Abujabal et al. 2017).

A variety of techniques have been adopted for text generation for a variety of inputs such as text, graphs and data records. Earlier attempts for text generation adopted a combination of handcrafted rules and manually engineered templates to generate the output text (Reiter et al. 2005). Later, researchers started focusing on automatically generating templates to replace the manual ones (Angeli, Liang, and Klein 2010). Such techniques were applied on discrete text representations, which are less efficient for characterizing text semantics. Quests for more efficient methods such as those using dense sentence representations resulted in the development of neural text-to-text generation models have been applied to machine translation (Bahdanau, Cho, and Bengio 2014) and summarization (Kurisinkel, Zhang, and Varma 2019).

Text generation methods from data records have been investigated for different datasets such as wikipedia infobox (Lebret, Grangier, and Auli 2016) and (Kurisinkel and Chen 2019). There were studies in the past which investigated and evaluated text coherence (Barzilay and Lapata 2008) and attempted to improve coherence during text generation. There are also studies which incorporated context based text representations for efficient text generation (Clark, Ji, and Smith 2018). Our work proposes a neural approach by characterizing the input knowledge graph representation specific to the sentential context for coherent text generation.

Text generated to describe information characterized in graphs with labeled edges (e.g., knowledge graphs or abstract meaning representation (AMR)) include (Koncel-Kedziorski et al. 2019; Song et al. 2018; Veličković et al. 2017). (Konstas et al. 2017) pioneered the first work on neural text generation from AMR graphs by pre-training with noisy parsed information. Other attempts are made to encode graphs using graph convolution network and attention en-

coders (Marcheggiani and Perez-Beltrachini 2018). A portion of the work use gated information flow between nodes to learn the node representation within the graphical context (Beck, Haffari, and Cohn 2018; Song et al. 2018). Most of such work is on generating a single sentence from an input AMR, which are rooted graphs with a fine-grained set of labels defined for a single sentence. Thus, only one sentence can be generated from an AMR graph.

Recently, Koncel-Kedziorski et al. (2019) started work on multi-sentence generation from knowledge graph inputs guided by the title text, where they introduced graph transformers for encoding the input. They used a bipartite representation to linearize the graph and generated the multi-sentence output as a single sequence. By contrast, our approach views a graph as a set of edges and learns the context representation of edges within the graph using self attention. We generate the output sentence-by-sentence to improve coherence. Our approach explicitly computes the representation of the input graph at each sentential context by selecting content for coherence and informativeness through the gated mechanism.

Task Definition

Given a knowledge graph, each edge $e \in E$ can be represented with a triplet (ne_i, l_k, ne_j) , where ne_i and ne_j are named entities in adjacent nodes and l_k is the label of the edge that connects them. The set of edges is able to construct the graph. The proposed task generates a sequence of sentences to represent the information contained in the input knowledge graph guided by a proposed title text (see Figure 1). We generate the output in a sentence by sentence order. At each time step of sentence generation, we generate the in sentence in two steps as listed below,

- Computed the representation of input graph corresponding to current time- step by properly by weighting content selected from original graph representation and previous graph representation.

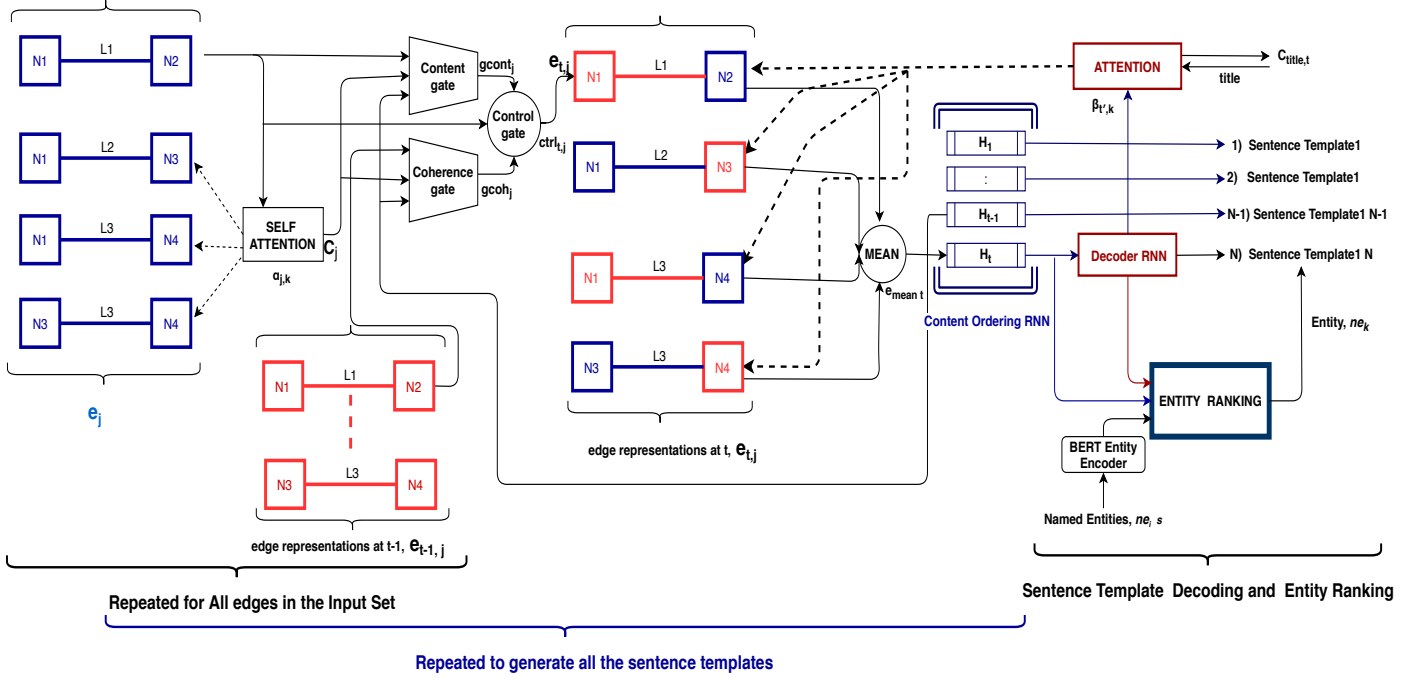


Figure 2: Proposed neural architecture: Initial edges representations are depicted in blue (e_j) and those for time -step $t - 1$ ($e_{t-1,j}$) in red. Edge representations computed for time- step t ($e_{t,j}$) are depicted using both red and blue as it encompasses content for coherence via topical continuity and informativeness via novelty

We present a \$ENTITY for Event Detection.
We evaluate this model on \$ENTITY Task.
Our Model outperforms \$ENTITY by 9% on this data

Figure 3: An example of a generated sequence of sentence templates

- Realize the sentence for the current time step from the computed time-step specific representation of the input graph, guided by the proposed title.

Approach

Our method first generates sentence templates and then fills the entity slots in the generated templates by ranking the input entities. The generated sequence of sentence templates for the given knowledge graph in Figure 1 is shown in Figure 3. Figure 2 shows the neural architecture of our proposed framework. The architecture comprises neural components for learning edge representation, gates for selecting content for coherence and informativeness, control gate mechanism for learning an optimum trade-off between coherence and informativeness, a decoder for individual sentence templates and an entity ranking scheme for ranking entities to fill slots in the templates. The following subsections elaborates on each of these components.

Edge Representation

Each edge in the graph is represented as a concatenation of adjacent node representations and the edge relation label representation. Specifically, representation e_q of edge (ne_i, l_k, ne_j) with ne_i and ne_j as entities at the nodes and edge label l_k is computed as the following:

$$\begin{aligned} ne'_i &= [\text{Ent}(\text{Key}(ne_i)), \text{E}_{\text{type}}(\text{typeof}(ne_i))] \\ ne'_j &= [\text{Ent}(\text{Key}(ne_j)), \text{E}_{\text{type}}(\text{typeof}(ne_j))] \\ e_q &= [ne'_i; E_r(l_k); ne'_j] \end{aligned}$$

Key represents the entity key of entity in the context of current input graph (eg: *task_1* is the key for the entity *EventDetection* in Figure 1), *Ent* is a look up table for entity keys, *typeof* represents the entity type of argument entity, *E_{type}* is the entity type look up table and *E_r* is the edge relation label look up table.

In a graph, edges cannot be treated as discrete elements, but should be defined in the context of the graph. For this purpose, our method computes the context vector for each edge which represents the context of the edge within the input graph. Steps involved in the self attention mechanism used for computing the context vector of edge e_j is done as follows.

$$\begin{aligned} \alpha_{j,k} &= \exp(e_j^T \mathbf{W}_a e_k) \\ C_j &= \sum_{k \neq j} \alpha_{j,k} e_k \end{aligned}$$

where \mathbf{W}_a is a parameter matrix.

Content Selection for Coherence and Informativeness

The output is generated in a sentence-by-sentence order. The content ordering RNN stores the current context of the output sentence generation. The content of the sentence to be generated at each stage is implemented by selection gates. There are two selection gates which select content for coherence via topical continuity and informativeness via novel content selection. The control gate weights determine the relative ratios between coherence and informativeness during each sentence template generation phase.

At the time step t of the content ordering RNN with current hidden state H_{t-1} , the content selection for generating template for sentence to be constructed is done using content and coherence gates. The computation process of H_{t-1} is explained in a subsequent section. At time-step t , for each edge e_i , the content gate $gcont$ selects content from the original edge representations as follows:

$$\begin{aligned} gcont_{t,j} &= \text{sigmoid}(\mathbf{W}_{gc}[e_j; C_j; H_{t-1}]) \\ econt_{t,j} &= gcont_{t,j} \odot e_j \end{aligned}$$

where \mathbf{W}_{gc} is a parameter matrix. The computation of e_j and C_j is explained in the previous section (Edge Representation). The coherence gate $gcoh$ selects the content for topical continuity from edge representations computed for previous time-step as follows,

$$\begin{aligned} gcoh_{t,j} &= \text{sigmoid}(\mathbf{W}_{gs}[e_{t-1,j}; C_j; H_{t-1}]) \\ ecoh_{t,j} &= gcoh_{t,j} \odot e_{t-1,j} \end{aligned}$$

where \mathbf{W}_{gs} is a parameter matrix and $e_{t-1,j}$ is an edge content representation computed at the previous time-step (Equation 1 with recursive computation). The coherence gate is inactive during the generation of the first sentence as there is no previous content. The control gate computes the weightage between the content for coherence and informativeness and computes the edge content representation for the time-step t as follows.

$$\begin{aligned} ctrl_{t,j} &= \text{sigmoid}(\mathbf{W}_{gc}[H_{t-1} + b]) \\ e_{t,j} &= ctrl_{t,j} * econt_{t,j} + (1 - ctrl_{t,j}) * ecoh_{t,j} \end{aligned} \quad (1)$$

During each stage of feature and edge selection, content ordering RNN state H_{t-1} provides feedback for gates regarding already covered information and topical context. This feedback enables the gates to compute content for the next sentence to be generated. In Figure 2, trapezoids represent the content and coherence gates while the left portion of the Figure illustrates the self attention mechanism of the neural network for computing C_j .

Content Ordering & Decoding Output Sentence Templates

Content order RNN, a GRU recurrent neural network, is initialized with a zero vector state before the first forward propagation of the network begins. Input at each time step t of the content ordering RNN is e_{mean_t} , the mean of all edge representations ($e_{t,j}, \forall j$). Therefore the hidden state H_t of the hidden state RNN is computed as follows:

$$H_t = \text{GRU}(e_{mean_t}, H_{t-1}), \quad (2)$$

Sentence template decoder RNN is a GRU and initialized with H_t , the current state of content ordering RNN to generate the sentence template corresponding to current time-step t . The template decoder RNN attends over the updated set of edges $e_j, \forall j$ to generate the sequence of tokens in the sentence template to be decoded. Also, the entire template generation is conditioned on the expected title of the text to be generated. The computations involved in the process is given below.

$$\begin{aligned} \beta_{t',k} &= \exp(h_{t'}^T \mathbf{W}_c e_{t,k}) \\ c_{t'} &= \sum \beta_{t',k} E_{t,k} \\ \gamma_k &= \exp(Y_{title}^T \mathbf{W}_c e_{t,k}) \\ c_{t',title} &= \sum \gamma_k e_{t,k} \\ h_{t'}' &= \tanh(\mathbf{W}_h[h_{t'}, c_{t'}, c_{t',title}]) \end{aligned}$$

where $h_{t'}$ is the hidden state of the decoder at time-step t' . Y_{title} is the representation of title text constructed using a pre-trained BERT model (Devlin et al. 2019). The probability distribution over the output vocabulary for generating the token $w_{t'}$ at time step t' of the template generating decoder is computed as below:

$$\begin{aligned} P(w_{t'} | w_{<t'}, \{e_i, \dots, e_n\}, X_{title}) &= \\ \text{softmax}(\mathbf{W}_o h_{t'}' + b_o) \end{aligned} \quad (3)$$

The rightmost portion of Figure 2 marked in dark blue and dark red represents the set of computations detailed in the current section. Figure 3 depicts the example of a generated sequence of sentence templates.

Entity Ranking

Once the sentence template is created, we rank entities for filling the entity slots created in the generated sentence templates. For an accurate ranking sentential and word contexts need to be considered. The hidden state of the content ordering RNN represents the content of the sentence under construction while that of the template decoder represents the current token to be generated. For ranking entities to fill the the template generated at time-step t of content ordering RNN, we take into account the hidden states of content ordering RNN and the template decoder hidden states. We compute the score for entity ne_i to fill the entity slot generated at time-step t' of template decoder as follows.

$$\begin{aligned} score(e_i) &= \mathbf{V}^t \tanh(\mathbf{W}_r[H_t, h_{t'}, ne_i']) \\ score(e_i) &= \text{softmax}(score(e_i)) \quad \forall i \end{aligned} \quad (4)$$

where \mathbf{V}^t and \mathbf{W}_r are parameter matrices and ne_i' is a dense representation of the entity ne_i computed out of its textual description (eg: *event detection* at the node $N2$ in Figure 1) using a pre-trained BERT model (Devlin et al. 2019). The particular slot in the sentence template generated is filled with the entity with maximum score.

Loss Function

The loss function incorporates components to maximize template sentence and template token generation accuracy and entity scoring for filling the generated sentence template.

$$\mathcal{C} = -\sum_{i=1}^T \sum_{j=1}^{t'} \log P_{i,j} - \sum_{i=1}^T \sum_{\forall \text{entity_slot}_j} \log (\text{Score}(e_{i,j})) \quad (5)$$

The first component of the loss function \mathcal{C} minimizes the error during sentence template generation, as $P_{i,j}$ is the probability of the expected token during decoding. T is content ordering time steps (i.e. number of sentences); t' is time steps of the decoder (i.e. number of word tokens). The second component minimizes the error of entity ranking, where the expected entity $e_{i,j}$ is computed in Equation (4). Loss is summed up for all the time steps of the content ordering RNN and the decoder RNN.

Experiments

Data

We use the data released by Koncel-Kedziorski et al. (2019) and followed the default data split for training and evaluation. They took 40K paper titles and abstracts from the Semantic Scholar Corpus (Ammar et al. 2018). From each of the abstract collected, a knowledge graph is constructed using SciIE (Luan et al. 2018), a system with established efficiency for information extraction in the scientific domain. SciIE was used to extract major named entities from the abstract, whereas these named entities belong to all categories in the scientific paper such as Methods and Material, Evaluation Metrics, or Experimental Results. SciIE was also used to identify the relationship between different entities mentioned in the abstract. The templates for template generation is constructed by replacing named entities in each output sentence in training data by an *\$Entity_slot*.

Model Comparison Settings

We compare different settings of our approach with other previously proposed approaches.

- GraphWriter: The approach proposed by Koncel-Kedziorski et al. (2019) which uses graph transformer to encode the input graph.
- GAT: This approach uses graph attention network to encode the input knowledge graph by attending over neighbouring edgess Velickovic et al. (2017).
- EntityWriter: This is a base model used by Koncel-Kedziorski et al. (2019) in which only entities and title text is used as input.
- Graph2Order : This is a variant of our approach in which content selection gate is present, but coherence and control gates are not present.
- Graph2CoherentText : This setting represents our approach proposed in the current paper with content selection, coherence and control gates.

Method	BLEU	METEOR
GraphWriter	14.3+/-1.01	18.8 +/- 0.28
GAT	12.2+/-0.44	17.2 +/- 0.63
EntityWriter	10.38	16.53
Graph2Order	14.20+/-0.7	18.20+/-0.3
Graph2CoherentText	15.40+/-0.9	19.30+/-0.19

Table 1: Evaluation I: Content Generation

Experimental Setup

Entity representation has a dimension of 300. Entity type and edge label representations both have a dimension of 33. Dimensions of network parameter matrices are tuned accordingly. BERT embeddings with a size of 768² pass through a linear transformation for adapting to the network computations. To train the network we used SGD with the momentum set to 0.9 and a learning rate of 0.01.

Evaluation Study

Evaluation I: Content Generation

We evaluated the quality of content generation using the BLEU (Papineni et al. 2002) and METEOR (Denkowski and Lavie 2014) metrics. The results are shown in Table 1. *Graph2CoherentText* consistently outperforms other approaches for both metrics while *Graph2Order* shows results comparable to *GraphWriter*, suggesting that graph encoding scheme used in these methods incorporating edge representations and edge context vectors, are equally efficient in capturing semantics of input graph. Both *Graph2CoherentText* and *GraphWriter* generate content in a sentence-by-sentence manner. However, *Graph2CoherentText* yields better results for both BLEU and METEOR, indicating that mere content order RNN alone is insufficient to model the topical context of a sentence. In *Graph2CoherentText*, topical continuity is ensured by allowing information from the previous edge representations to penetrate through the coherence gate, resulting in informative content creation for a new sentence where novelty and topical continuity are weighed by the control gate.

Evaluation II: Human Evaluation

We randomly choose 30 knowledge graphs in the test set and the corresponding text generated by each of the competing approaches. The pairs of graphs and output text are randomly presented to the evaluators. The generated text is evaluated on the dimensions of coherence, informativeness and grammaticality. The human evaluators are 4 post-graduate students in linguistics. For coherence, evaluators are instructed to look for topical continuity between neighboring sentences. For Informativeness, evaluators are asked to decide how much of information contained in the input graph is covered in the generated text. For grammaticality, the evaluators are asked to judge the linguistic quality of generated sentences irrespective of information contained.

²<https://github.com/google-research/bert>

Method	Coherence	Informativeness	Grammaticality
GraphWriter	20%	27%	20%
Graph2Order	25%	26%	19%
Graph2CoherentText	45%	40%	27%
ambiguous	10%	7%	34%
Inter Evaluator Agreement (Kappa)	0.75	0.73	0.72

Table 2: Evaluation II: Human Evaluation Results

Title	Learning Reliability of Parsers for Domain Adaptation of Dependency Parsing
Input Graph	<pre> graph TD parsing((parsing)) -- EVALUATE-FOR --> relation_extraction((relation extraction)) parsing -- EVALUATE-FOR --> nlp_applications((natural language processing applications)) parsing -- EVALUATE-FOR --> paraphrase_acquisition((paraphrase acquisition)) dependency_parser((dependency parser)) -- USED_FOR --> parsing dependency_parser -- USED_FOR --> conll_task((conll 2007 shared task)) relation_extraction -- HYPONYM-OF --> nlp_applications relation_extraction -- HYPONYM-OF --> paraphrase_acquisition nlp_applications -- HYPONYM-OF --> paraphrase_acquisition </pre>
Reference Text	<p>the accuracy of parsing has exceeded 90 % recently , but this is not high enough to use parsing results practically in natural language processing applications such as paraphrase acquisition and relation extraction . we present a method for detecting reliable parses out of the outputs of a single dependency parser . this technique is also applied to domain adaptation of dependency parsing . our goal was to improve the performance of a state-of-the-art dependency parser on the data set of the domain adaptation track of the CoNll 2007 shared task , a formidable challenge .</p>
Output Text (proposed model)	<p>Parsing is effective in improving natural language processing applications such as paraphrase acquisition and relation extraction. We propose a novel method for parsing which is used for natural language processing. Performance of the dependency parser is evaluated on conll 2007 task...</p>

Table 3: An example of an automatically generated piece of text using the proposed approach.

Table 2 shows the distribution of the best performing model determined by the evaluators for each evaluation criteria (coherence, informativeness, grammaticality). *Graph2CoherentText* is overwhelmingly chosen by human evaluators for coherence and informativeness, suggesting the effectiveness of incorporating explicit means for coherence modeling through topical continuity and content selection at each sentential context. No approach yielded a significant dominance in the evaluation for grammaticality, which could be because given the dataset, all models are equipped to do language modelling reasonably well irrespective of whether the content is informative or not. Inter-evaluator agreement is quantified using Cohen’s kappa coefficient, and is at least 0.72 for each dimension of co-

herence, informativeness and grammaticality (Wongpakaran et al. 2013), as shown in Table 2.

Discussion

Table 3 shows an example where the automatically generated output text from using the proposed model from a relatively smaller knowledge graph. The information in the input graph are well-represented in the output text, and the output text is much more concise than the reference text. There is a coherent transition of topics between sentences with a full coverage of information represented in the knowledge graph. Note that for adjectives like *novel*, which appear a lot in methods research in scientific writing, the proposed

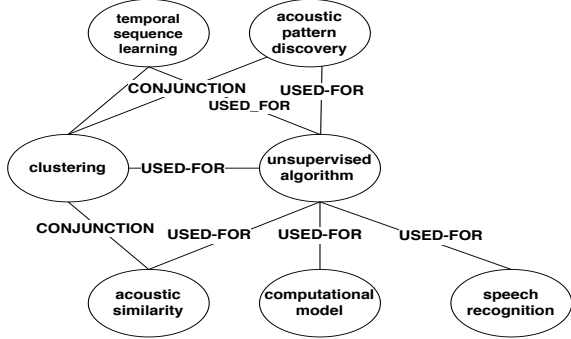
Title	A computational model for unsupervised word discovery
Input Graph	
Reference Text	<p>we present an unsupervised algorithm for the discovery of words and word-like fragments from the speech signal , without using an upfront defined lexicon or acoustic phone models <i>first , the unsupervised algorithm may lead to an approach for speech recognition that is fundamentally liberated from the modelling constraints in conventional automatic speech recognition</i> . second , the proposed unsupervised algorithm can be interpreted as a computational model of language acquisition that takes actual speech as input and is able to find words as 'em emergent ' properties from raw input .we present a novel approach</p>

Table 4: An example where the input graph does not contain sufficient information for generating the reference text. Portions that cannot be inferred from the input graph are in blue.

model is able to learn when to use it appropriately.

However, on the other hand, for less common adjectives like *formidable*, which require more technical background knowledge, the automatic model is less likely to learn how to generate it appropriately in the given context. To further illustrate this issue, Table 4 shows an example where there is more content in the reference that cannot be directly reconstructed from the input. For example, consider the sentence highlighted in blue in Table 4: *first , the unsupervised algorithm may lead to an approach for speech recognition that is fundamentally liberated from the modelling constraints in conventional automatic speech recognition*. This sentence cannot be constructed directly from the input knowledge graph, as it does not encompass any domain knowledge about conventional approaches in speech recognition. Such could be a possible reason why the BLEU performance is relatively low for all systems.

Conclusion

In this work, we proposed a neural modeling framework for generating coherent text given a knowledge graph, where we model content ordering through topic continuity while maintaining comprehensive information coverage represented via exploiting graph representations specific to each sentential context. In addition, we also adopted a sentence-by-sentence decoding scheme, which improves tractability of long sequence generation. We empirically verified our proposed framework on the AGENDA dataset. Our proposed approach enhanced both sequence generation order and in-

formation coverage using objective metrics such as BLEU and METEOR in comparison to established baselines using graph transformer networks. Human evaluations also showed our approach is especially effective on generating informative and coherent text descriptions from the given knowledge graph.

References

- Abujabal, A.; Yahya, M.; Riedewald, M.; and Weikum, G. 2017. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th international conference on world wide web*, 1191–1200. International World Wide Web Conferences Steering Committee.
- Ammar, W.; Groeneveld, D.; Bhagavatula, C.; Beltagy, I.; Crawford, M.; Downey, D.; Dunkelberger, J.; Elgohary, A.; Feldman, S.; Ha, V.; et al. 2018. Construction of the literature graph in semantic scholar. *arXiv preprint arXiv:1805.02262*.
- Angeli, G.; Liang, P.; and Klein, D. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 502–512. Association for Computational Linguistics.
- Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Bao, J.; Duan, N.; Yan, Z.; Zhou, M.; and Zhao, T. 2016.

- Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2503–2514.
- Barzilay, R.; and Lapata, M. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics* 34(1): 1–34.
- Beck, D.; Haffari, G.; and Cohn, T. 2018. Graph-to-sequence learning using gated graph neural networks. *arXiv preprint arXiv:1806.09835*.
- Clark, E.; Ji, Y.; and Smith, N. A. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2250–2260.
- Denkowski, M.; and Lavie, A. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, 376–380.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT (1)*.
- Kasneci, G.; Elbassuoni, S.; and Weikum, G. 2009. Ming: mining informative entity relationship subgraphs. In *Proceedings of the 18th ACM conference on Information and knowledge management*, 1653–1656. ACM.
- Koncel-Kedziorski, R.; Bekal, D.; Luan, Y.; Lapata, M.; and Hajishirzi, H. 2019. Text Generation from Knowledge Graphs with Graph Transformers. *arXiv preprint arXiv:1904.02342*.
- Konstas, I.; Iyer, S.; Yatskar, M.; Choi, Y.; and Zettlemoyer, L. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *arXiv preprint arXiv:1704.08381*.
- Kurisinkel, L. J.; and Chen, N. 2019. Set to Ordered Text: Generating Discharge Instructions from Medical Billing Codes. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 6166–6176.
- Kurisinkel, L. J.; Zhang, Y.; and Varma, V. 2019. Domain Adaptive Neural Sentence Compression by Tree Cutting. In *European Conference on Information Retrieval*, 475–488. Springer.
- Lebret, R.; Grangier, D.; and Auli, M. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1203–1213.
- Luan, Y.; He, L.; Ostendorf, M.; and Hajishirzi, H. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602*.
- Marcheggiani, D.; and Perez-Beltrachini, L. 2018. Deep graph convolutional encoders for structured data to text generation. *arXiv preprint arXiv:1810.09995*.
- Mehta, A.; Singhal, A.; and Karlapalem, K. 2019. Scalable Knowledge Graph Construction over Text using Deep Learning based Predicate Mapping. In *Companion Proceedings of The 2019 World Wide Web Conference*, 705–713. ACM.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318. Association for Computational Linguistics.
- Pu, M.-M. 2006. Discourse organization and coherence. *Cognitive Linguistics Investigations: Across Languages, Fields and Philosophical Boundaries* 305.
- Reiter, E.; Sripada, S.; Hunter, J.; Yu, J.; and Davy, I. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence* 167(1-2): 137–169.
- Song, L.; Zhang, Y.; Wang, Z.; and Gildea, D. 2018. A Graph-to-Sequence Model for AMR-to-Text Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1616–1626.
- Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.
- Wang, H.; Zhang, F.; Xie, X.; and Guo, M. 2018. DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, 1835–1844. International World Wide Web Conferences Steering Committee.
- Wongpakaran, N.; Wongpakaran, T.; Wedding, D.; and Gwet, K. L. 2013. A comparison of Cohen’s Kappa and Gwet’s AC1 when calculating inter-rater reliability coefficients: a study conducted with personality disorder samples. *BMC medical research methodology* 13(1): 61.
- Zhang, Y.; Dai, H.; Kozareva, Z.; Smola, A. J.; and Song, L. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.