Semantic Graphs for Generating Deep Questions

Liangming Pan[1,2] **Yuxi Xie**[3] **Yansong Feng**[3]
Tat-Seng Chua[2] **Min-Yen Kan**[2]
[1]NUS Graduate School for Integrative Sciences and Engineering
[2]School of Computing, National University of Singapore, Singapore
[3]Wangxuan Institute of Computer Technology, Peking University
e0272310@u.nus.edu, {xieyuxi, fengyansong}@pku.edu.cn
{dcscts@, kanmy@comp.}nus.edu.sg

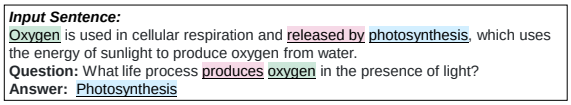
Abstract

This paper proposes the problem of Deep Question Generation (DQG), which aims to generate complex questions that require reasoning over multiple pieces of information of the input passage. In order to capture the global structure of the document and facilitate reasoning, we propose a novel framework which first constructs a semantic-level graph for the input document and then encodes the semantic graph by introducing an attention-based GGNN (Att-GGNN). Afterwards, we fuse the document-level and graph-level representations to perform joint training of content selection and question decoding. On the HotpotQA deep-question centric dataset, our model greatly improves performance over questions requiring reasoning over multiple facts, leading to state-of-the-art performance. The code is publicly available at https://github.com/WING-NUS/SG-Deep-Question-Generation.

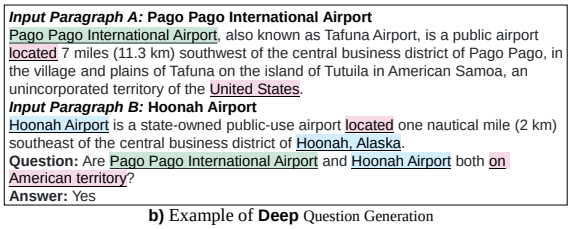
1 Introduction

Question Generation (QG) systems play a vital role in question answering (QA), dialogue system, and automated tutoring applications – by enriching the training QA corpora, helping chatbots start conversations with intriguing questions, and automatically generating assessment questions, respectively. Existing QG research has typically focused on generating factoid questions relevant to one fact obtainable from a single sentence (Duan et al., 2017; Zhao et al., 2018; Kim et al., 2019), as exemplified in Figure 1 a). However, less explored has been the comprehension and reasoning aspects of questioning, resulting in questions that are shallow and not reflective of the true creative human process.

People have the ability to ask deep questions about events, evaluation, opinions, synthesis, or reasons, usually in the form of *Why*, *Why-not*, *How*, *What-if*, which requires an in-depth understanding of the input source and the ability to reason over disjoint relevant contexts; *e.g.*, asking *Why did Gollum betray his master Frodo Baggins?* after reading the fantasy novel *The Lord of the Rings*. Learning to ask such deep questions has intrinsic research value concerning how human intelligence embodies the skills of curiosity and integration, and will have broad application in future intelligent systems. Despite a clear push towards *answering* deep questions (exemplified by multi-hop reading comprehension (Cao et al., 2019) and commonsense QA (Rajani et al., 2019)), *generating* deep questions remains un-investigated. There is thus a clear need to push QG research towards generating deep questions that demand higher cognitive skills.



a) Example of **Shallow** Question Generation



b) Example of **Deep** Question Generation

Figure 1: Examples of shallow/deep QG. The evidence needed to generate the question are highlighted.

In this paper, we propose the problem of **D**eep **Q**uestion **G**eneration (DQG), which aims to generate questions that require reasoning over multiple pieces of information in the passage. Figure 1 b) shows an example of deep question which requires a comparative reasoning over two disjoint pieces of evidences. DQG introduces three additional challenges that are not captured by traditional QG systems. First, unlike generating questions from a single sentence, DQG requires document-level

long dependency

understanding, which may introduce long-range dependencies when the passage is long. Second, we must be able to select relevant contexts to ask meaningful questions; this is non-trivial as it involves understanding the relation between disjoint pieces of information in the passage. Third, we need to ensure correct reasoning over multiple pieces of information so that the generated question is answerable by information in the passage.

To facilitate the selection and reasoning over disjoint relevant contexts, we distill important information from the passage and organize them as a *semantic graph*, in which the nodes are extracted based on semantic role labeling or dependency parsing, and connected by different intra- and inter-semantic relations (Figure 2). Semantic relations provide important clues about what contents are question-worthy and what reasoning should be performed; *e.g.*, in Figure 1, both the entities *Pago Pago International Airport* and *Hoonah Airport* have the *located_at* relation with a city in United States. It is then natural to ask a comparative question: *e.g.*, *Are Pago Pago International Airport and Hoonah Airport both on American territory?*. To efficiently leverage the semantic graph for DQG, we introduce three novel mechanisms: (1) proposing a novel graph encoder, which incorporates an attention mechanism into the Gated Graph Neural Network (GGNN) (Li et al., 2016), to dynamically model the interactions between different semantic relations; (2) enhancing the word-level passage embeddings and the node-level semantic graph representations to obtain an unified semantic-aware passage representations for question decoding; and (3) introducing an auxiliary *content selection* task that jointly trains with question decoding, which assists the model in selecting relevant contexts in the semantic graph to form a proper reasoning chain.

We evaluate our model on HotpotQA (Yang et al., 2018), a challenging dataset in which the questions are generated by reasoning over text from separate Wikipedia pages. Experimental results show that our model — incorporating both the use of the semantic graph and the content selection task — improves performance by a large margin, in terms of both automated metrics (Section 4.3) and human evaluation (Section 4.5). Error analysis (Section 4.6) validates that our use of the semantic graph greatly reduces the amount of semantic errors in generated questions. In summary, our contributions are: (1) the very first work, to the best of our knowledge, to investigate deep question generation, (2) a novel framework which combines a semantic graph with the input passage to generate deep questions, and (3) a novel graph encoder that incorporates attention into a GGNN approach.

2 Related Work

Question generation aims to automatically generate questions from textual inputs. *Rule-based techniques* for QG usually rely on manually-designed rules or templates to transform a piece of given text to questions (Heilman, 2011; Chali and Hasan, 2012). These methods are confined to a variety of transformation rules or templates, making the approach difficult to generalize. *Neural-based approaches* take advantage of the sequence-to-sequence (Seq2Seq) framework with attention (Bahdanau et al., 2014). These models are trained in an end-to-end manner, requiring far less labor and enabling better language flexibility, compared against rule-based methods. A comprehensive survey of QG can be found in Pan et al. (2019).

Many improvements have been proposed since the first Seq2Seq model of Du et al. (2017): applying various techniques to encode the answer information, thus allowing for better quality answer-focused questions (Zhou et al., 2017; Sun et al., 2018; Kim et al., 2019); improving the training via combining supervised and reinforcement learning to maximize question-specific rewards (Yuan et al., 2017); and incorporating various linguistic features into the QG process (Liu et al., 2019a). However, these approaches only consider sentence-level QG. In contrast, our work focus on the challenge of generating deep questions with multi-hop reasoning over document-level contexts.

Recently, work has started to leverage paragraph-level contexts to produce better questions. Du and Cardie (2018) incorporated coreference knowledge to better encode entity connections across documents. Zhao et al. (2018) applied a gated self-attention mechanism to encode contextual information. However, in practice, semantic structure is difficult to distil solely via self-attention over the entire document. Moreover, despite considering longer contexts, these works are trained and evaluated on SQuAD (Rajpurkar et al., 2016), which we argue as insufficient to evaluate deep QG because more than 80% of its questions are shallow and only relevant to information confined to a single sentence (Du et al., 2017).

对我们的simplification有没有帮助

coherence

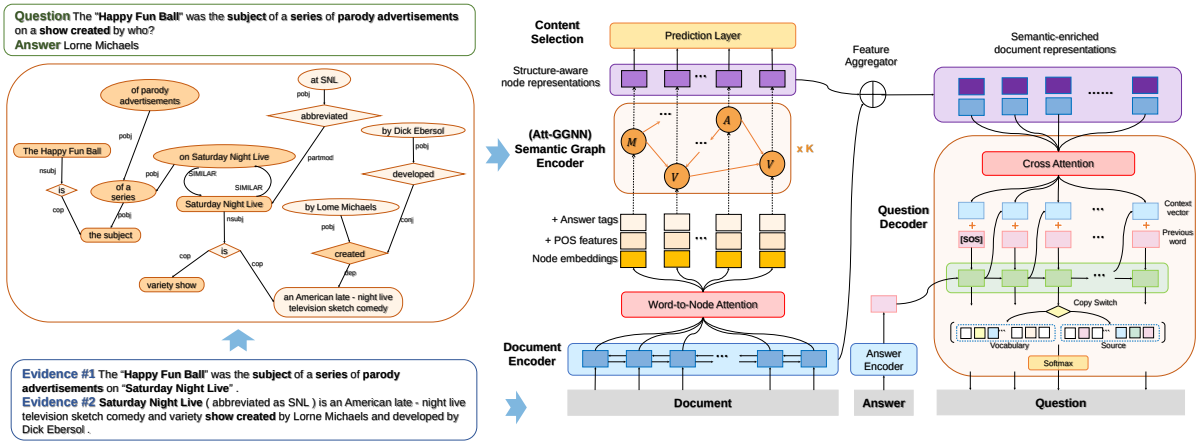


Figure 2: The framework of our proposed model (on the right) together with an input example (on the left). The model consists of four parts: (1) a *document encoder* to encode the input document, (2) a *semantic graph encoder* to embed the document-level semantic graph via Att-GGNN, (3) a *content selector* to select relevant question-worthy contents from the semantic graph, and (4) a *question decoder* to generate question from the semantic-enriched document representation. The left figure shows an input example and its semantic graph. Dark-colored nodes in the semantic graph are question-worthy nodes that are labeled to train the content selection task.

3 Methodology

Given the document $\mathcal{D}$ and the answer $\mathcal{A}$, the objective is to generate a question $\bar{\mathcal{Q}}$ that satisfies:

$$\bar{\mathcal{Q}} = \arg\max_{\mathcal{Q}} P(\mathcal{Q}|\mathcal{D}, \mathcal{A}) \quad (1)$$

where document $\mathcal{D}$ and answer $\mathcal{A}$ are both sequences of words. Different from previous works, we aim to generate a $\bar{\mathcal{Q}}$ which involves reasoning over multiple evidence sentences $\mathcal{E} = \{s_i\}_{i=1}^{n}$, where s_i is a sentence in $\mathcal{D}$. Also, unlike traditional settings, $\mathcal{A}$ may not be a sub-span of $\mathcal{D}$ because reasoning is involved to obtain the answer.

3.1 General Framework

We propose an encoder–decoder framework with two novel features specific to DQG: (1) a fused word-level document and node-level semantic graph representation to better utilize and aggregate the semantic information among the relevant disjoint document contexts, and (2) joint training over the question decoding and content selection tasks to improve selection and reasoning over relevant information. Figure 2 shows the general architecture of the proposed model, including three modules: *semantic graph construction*, which builds the DP- or SRL-based semantic graph for the given input; *semantic-enriched document representation*, employing a novel Attention-enhanced Gated Graph Neural Network (Att-GGNN) to learn the semantic graph representations, which are then fused with the input document to obtain graph-enhanced document representations; and *joint-task question generation*, which generates deep questions via joint training of node-level content selection and word-level question decoding. In the following, we describe the details of each module.

3.2 Semantic Graph Construction

As illustrated in the introduction, the semantic relations between entities serve as strong clues in determining what to ask about and the reasoning types it includes. To distill such semantic information in the document, we explore both SRL- (Semantic Role Labelling) and DP- (Dependency Parsing) based methods to construct the semantic graph. Refer to Appendix A for the details of graph construction.

• **SRL-based Semantic Graph.** The task of Semantic Role Labeling (SRL) is to identify what semantic relations hold among a predicate and its associated participants and properties (Màrquez et al., 2008), including "who" did "what" to "whom", etc. For each sentence, we extract predicate-argument tuples via SRL toolkits[1]. Each tuple forms a subgraph where each tuple element (*e.g.*, arguments, location, and temporal) is a node. We add inter-tuple edges between nodes from different tuples if they have an inclusive relationship or potentially mention the same entity.

[1]We employ the state-of-the-art BERT-based model (Shi and Lin, 2019) in the AllenNLP toolkit to perform SRL.

• **DP-based Semantic Graph.** We employ the biaffine attention model (Dozat and Manning, 2017) for each sentence to obtain its dependency parse tree, which is further revised by removing unimportant constituents (*e.g.*, punctuation) and merging consecutive nodes that form a complete semantic unit. Afterwards, we add inter-tree edges between similar nodes from different parse trees to construct a connected semantic graph.

The left side of Figure 2 shows an example of the DP-based semantic graph. Compared with SRL-based graphs, DP-based ones typically model more fine-grained and sparse semantic relations, as discussed in Appendix A.3. Section 4.3 gives a performance comparison on these two formalisms.

3.3 Semantic-Enriched Document Representations

We separately encode the document $\mathcal{D}$ and the semantic graph $\mathcal{G}$ via an RNN-based passage encoder and a novel Att-GGNN graph encoder, respectively, then fuse them to obtain the semantic-enriched document representations for question generation.

Document Encoding. Given the input document $\mathcal{D} = [w_1, \cdots, w_l]$, we employ the bi-directional Gated Recurrent Unit (GRU) (Cho et al., 2014) to encode its contexts. We represent the encoder hidden states as $\mathbf{X}_\mathcal{D} = [\mathbf{x}_1, \cdots, \mathbf{x}_l]$, where $\mathbf{x}_i = [\overrightarrow{\mathbf{x}_i}; \overleftarrow{\mathbf{x}_i}]$ is the context embedding of w_i as a concatenation of its bi-directional hidden states.

Node Initialization. We define the SRL- and DP-based semantic graphs in an unified way. The semantic graph of the document $\mathcal{D}$ is a heterogeneous multi-relation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_i\}_{i=1:N^v}$ and $\mathcal{E} = \{e_k\}_{i=1:N^e}$ denote graph nodes and the edges connecting them, where N^v and N^e are the numbers of nodes and edges in the graph, respectively. Each node $v = \{w_j\}_{j=m_v}^{n_v}$ is a text span in $\mathcal{D}$ with an associated node type t_v, where m_v / n_v is the starting / ending position of the text span. Each edge also has a type t_e that represents the semantic relation between nodes.

We obtain the initial representation $\mathbf{h}_v^0$ for each node $v = \{w_j\}_{j=m_v}^{n_v}$ by computing the word-to-node attention. First, we concatenate the last hidden states of the document encoder in both directions as the document representation $\mathbf{d}_\mathcal{D} = [\overrightarrow{\mathbf{x}_l}; \overleftarrow{\mathbf{x}_1}]$. Afterwards, for a node v, we calculate the attention distribution of $\mathbf{d}_\mathcal{D}$ over all the words $\{w_{m_v}, \cdots, w_j, \cdots, w_{n_v}\}$ in v as follows:

$$\beta_j^v = \frac{\exp(\text{Attn}(\mathbf{d}_\mathcal{D}, \mathbf{x}_j))}{\sum_{k=m_n}^{n_v} \exp(\text{Attn}(\mathbf{d}_\mathcal{D}, \mathbf{x}_k))} \tag{2}$$

where β_j^v is the attention coefficient of the document embedding $\mathbf{d}_\mathcal{D}$ over a word w_j in the node v. The initial node representation $\mathbf{h}_v^0$ is then given by the attention-weighed sum of the embeddings of its constituent words, *i.e.*, $\mathbf{h}_v^0 = \sum_{k=m_v}^{n_v} \beta_j^v \mathbf{x}_j$. Word-to-node attention ensures each node to capture not only the meaning of its constituting part but also the semantics of the entire document. The node representation is then enhanced with two additional features: the POS embedding $\mathbf{p}_v$ and the answer tag embedding $\mathbf{a}_v$ to obtain the enhanced initial node representations $\tilde{\mathbf{h}}_v^0 = [\mathbf{h}_v^0; \mathbf{p}_v; \mathbf{a}_v]$.

Graph Encoding. We then employ a novel Att-GGNN to update the node representations by aggregating information from their neighbors. To represent multiple relations in the edge, we base our model on the multi-relation Gated Graph Neural Network (GGNN) (Li et al., 2016), which provides a separate transformation matrix for each edge type. For DQG, it is essential for each node to pay attention to different neighboring nodes when performing different types of reasoning. To this end, we adopt the idea of Graph Attention Networks (Velickovic et al., 2017) to dynamically determine the weights of neighboring nodes in message passing using an attention mechanism.

Formally, given the initial hidden states of graph $\mathbf{H}^0 = \{\tilde{\mathbf{h}}_i^0\}|_{v_i \in \mathcal{V}}$, Att-GGNN conducts K layers of state transitions, leading to a sequence of graph hidden states $\mathbf{H}^0, \mathbf{H}^1, \cdots, \mathbf{H}^K$, where $\mathbf{H}^k = \{\mathbf{h}_j^{(k)}\}|_{v_j \in \mathcal{V}}$. At each state transition, an aggregation function is applied to each node v_i to collect messages from the nodes directly connected to v_i. The neighbors are distinguished by their incoming and outgoing edges as follows:

$$\mathbf{h}_{\mathcal{N}_\vdash(i)}^{(k)} = \sum_{v_j \in \mathcal{N}_\vdash(i)} \alpha_{ij}^{(k)} \mathbf{W}^{t_{e_{ij}}} \mathbf{h}_j^{(k)} \tag{3}$$

$$\mathbf{h}_{\mathcal{N}_\dashv(i)}^{(k)} = \sum_{v_j \in \mathcal{N}_\dashv(i)} \alpha_{ij}^{(k)} \mathbf{W}^{t_{e_{ji}}} \mathbf{h}_j^{(k)} \tag{4}$$

where $\mathcal{N}_\dashv(i)$ and $\mathcal{N}_\vdash(i)$ denote the sets of incoming and outgoing edges of v_i, respectively. $\mathbf{W}^{t_{e_{ij}}}$ denotes the weight matrix corresponding to the edge type $t_{e_{ij}}$ from v_i to v_j, and $\alpha_{ij}^{(k)}$ is the attention

coefficient of v_i over v_j, derived as follows:

$$\alpha_{ij}^{(k)} = \frac{\exp\left(\text{Attn}(\mathbf{h}_i^{(k)}, \mathbf{h}_j^{(k)})\right)}{\sum_{t \in \mathcal{N}_{(i)}} \exp(\text{Attn}(\mathbf{h}_i^{(k)}, \mathbf{h}_t^{(k)}))} \quad (5)$$

where $\text{Attn}(\cdot, \cdot)$ is a single-layer neural network implemented as $\mathbf{a}^T[\mathbf{W}^A\mathbf{h}_i^{(k)}; \mathbf{W}^A\mathbf{h}_j^{(k)}]$, here $\mathbf{a}$ and $\mathbf{W}^A$ are learnable parameters. Finally, an GRU is used to update the node state by incorporating the aggregated neighboring information.

$$\mathbf{h}_i^{(k+1)} = \text{GRU}(\mathbf{h}_i^{(k)}, \left[\mathbf{h}_{\mathcal{N}_{\vdash(i)}}^{(k)} ; \mathbf{h}_{\mathcal{N}_{\dashv(i)}}^{(k)}\right]) \quad (6)$$

After the K-th state transition, we denote the final structure-aware representation of node v as $\mathbf{h}_v^K$.

Feature Aggregation. Finally, we fuse the semantic graph representations $\mathbf{H}^K$ with the document representations $\mathbf{X}_\mathcal{D}$ to obtain the semantic-enriched document representations $\mathbf{E}_\mathcal{D}$ for question decoding, as follows:

$$\mathbf{E}_\mathcal{D} = \text{Fuse}(\mathbf{X}_\mathcal{D}, \mathbf{H}^K) \quad (7)$$

We employ a simple matching-based strategy for the feature fusion function Fuse. For a word $w_i \in \mathcal{D}$, we match it to the smallest granularity node that contains the word w_i, denoted as $v_{M(i)}$. We then concatenate the word representation $\mathbf{x}_i$ with the node representation $\mathbf{h}_{v_{M(i)}}^K$, *i.e.*, $\mathbf{e}_i = [\mathbf{x}_i ; \mathbf{h}_{v_{M(i)}}^K]$. When there is no corresponding node $v_{M(i)}$, we concatenate $\mathbf{x}_i$ with a special vector close to $\vec{0}$.

The semantic-enriched representation $\mathbf{E}_\mathcal{D}$ provides the following important information to benefit question generation: (1) *semantic information*: the document incorporates semantic information explicitly through concatenating with semantic graph encoding; (2) *phrase information*: a phrase is often represented as a single node in the semantic graph (*cf* Figure 2 as an example); therefore its constituting words are aligned with the same node representation; (3) *keyword information*: a word (*e.g.*, a preposition) not appearing in the semantic graph is aligned with the special node vector mentioned before, indicating the word does not carry important information.

3.4 Joint Task Question Generation

Based on the semantic-rich input representations, we generate questions via jointly training on two tasks: *Question Decoding* and *Content Selection*.

Question Decoding. We adopt an attention-based GRU model (Bahdanau et al., 2014) with copying (Gu et al., 2016; See et al., 2017) and coverage mechanisms (Tu et al., 2016) as the question decoder. The decoder takes the semantic-enriched representations $\mathbf{E}_\mathcal{D} = \{\mathbf{e}_i, \forall w_i \in \mathcal{D}\}$ from the encoders as the attention memory to generate the output sequence one word at a time. To make the decoder aware of the answer, we use the average word embeddings in the answer to initialize the decoder hidden states.

At each decoding step t, the model learns to attend over the input representations $\mathbf{E}_\mathcal{D}$ and compute a context vector $\mathbf{c}_t$ based on $\mathbf{E}_\mathcal{D}$ and the current decoding state $\mathbf{s}_t$. Next, the copying probability $P_{cpy} \in [0, 1]$ is calculated from the context vector $\mathbf{c}_t$, the decoder state $\mathbf{s}_t$ and the decoder input y_{t-1}. P_{cpy} is used as a soft switch to choose between generating from the vocabulary, or copying from the input document. Finally, we incorporate the coverage mechanisms (Tu et al., 2016) to encourage the decoder to utilize diverse components of the input document. Specifically, at each step, we maintain a coverage vector $\mathbf{cov}_t$, which is the sum of attention distributions over all previous decoder steps. A coverage loss is computed to penalize repeatedly attending to the same locations of the input document.

Content Selection. To raise a deep question, humans select and reason over relevant content. To mimic this, we propose an auxiliary task of content selection to jointly train with question decoding. We formulate this as a node classification task, *i.e.*, deciding whether each node should be involved in the process of asking, *i.e.*, appearing in the reasoning chain for raising a deep question, exemplified by the dark-colored nodes in Figure 2.

To this end, we add one feed-forward layer on top of the final-layer of the graph encoder, taking the output node representations $\mathbf{H}^K$ for classification. We deem a node as positive ground-truth to train the content selection task if its contents appears in the ground-truth question or act as a bridge entity between two sentences.

Content selection helps the model to identify the question-worthy parts that form a proper reasoning chain in the semantic graph. This synergizes with the question decoding task which focuses on the fluency of the generated question. We jointly train these two tasks with weight sharing on the input representations.

4 Experiments

4.1 Data and Metrics

To evaluate the model's ability to generate deep questions, we conduct experiments on HotpotQA (Yang et al., 2018), containing ∼100K crowd-sourced questions that require reasoning over separate Wikipedia articles. Each question is paired with two supporting documents that contain the evidence necessary to infer the answer. In the DQG task, we take the supporting documents along with the answer as inputs to generate the question. However, state-of-the-art semantic parsing models have difficulty in producing accurate semantic graphs for very long documents. We therefore pre-process the original dataset to select relevant sentences, *i.e.*, the evidence statements and the sentences that overlap with the ground-truth question, as the input document. We follow the original data split of HotpotQA to pre-process the data, resulting in 90,440 / 6,072 examples for training and evaluation, respectively.

Following previous works, we employ BLEU 1–4 (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007), and ROUGE-L (Lin, 2004) as automated evaluation metrics. BLEU measures the average n-gram overlap on a set of reference sentences. Both METEOR and ROUGE-L specialize BLEU's n-gram overlap idea for machine translation and text summarization evaluation, respectively. Critically, we also conduct human evaluation, where annotators evaluate the generation quality from three important aspects of deep questions: fluency, relevance, and complexity.

4.2 Baselines

We compare our proposed model against several strong baselines on question generation.

• **Seq2Seq + Attn** (Bahdanau et al., 2014): the basic Seq2Seq model with attention, which takes the document as input to decode the question.
• **NQG++** (Zhou et al., 2017): which enhances the Seq2Seq model with a feature-rich encoder containing answer position, POS and NER information.
• **ASs2s** (Kim et al., 2019): learns to decode questions from an answer-separated passage encoder together with a keyword-net based answer encoder.
• **S2sa-at-mp-gsa** (Zhao et al., 2018): an enhanced Seq2Seq model incorporating gated self-attention and maxout-pointers to encode richer passage-level contexts (B4 in Table 1). We also implement a version that uses coverage mechanism and our answer encoder for fair comparison, labeled B5.
• **CGC-QG** (Liu et al., 2019a): another enhanced Seq2Seq model that performs word-level content selection before generation; *i.e.*, making decisions on which words to generate and to copy using rich syntactic features, such as NER, POS, and DEP.

Implementation Details. For fair comparison, we use the original implementations of ASs2s and CGC-QG to apply them on HotpotQA. All baselines share a 1-layer GRU document encoder and question decoder with hidden units of 512 dimensions. Word embeddings are initialized with 300-dimensional pre-trained GloVe (Pennington et al., 2014). For the graph encoder, the node embedding size is 256, plus the POS and answer tag embeddings with 32-D for each. The number of layers K is set to 3 and hidden state size is 256. Other settings for training follow standard best practice[2].

4.3 Comparison with Baseline Models

The top two parts of Table 1 show the experimental results comparing against all baseline methods. We make three main observations:

1. The two versions of our model — P1 and P2 — consistently outperform all other baselines in BLEU. Specifically, our model with DP-based semantic graph (P2) achieves an absolute improvement of 2.05 in BLEU-4 (+15.2%), compared to the document-level QG model which employs gated self-attention and has been enhanced with the same decoder as ours (B5). This shows the significant effect of semantic-enriched document representations, equipped with auxiliary content selection for generating deep questions.

2. The results of CGC-QG (B6) exhibits an unusual pattern compared with other methods, achieving the *best* METEOR and ROUGE-L but *worst* BLEU-1 among all baselines. As CGC-QG performs word-level content selection, we observe that it tends to include many irrelevant words in the question, leading to lengthy questions (33.7 tokens on average, while 17.7 for ground-truth questions and 19.3 for our model) that are unanswerable or with semantic errors. Our model greatly reduces the error with node-level content selection based on semantic relations (shown in Table 3).

[2]All models are trained using Adam (Kingma and Ba, 2015) with mini-batch size 32. The learning rate is initially set to 0.001, and adaptive learning rate decay applied. We adopt early stopping and the dropout rate is set to 0.3 for both encoder and decoder and 0.1 for all attention mechanisms.

Model		BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE-L
Baselines	B1. Seq2Seq + Attn	32.97	21.11	15.41	11.81	18.19	33.48
	B2. NQG++	35.31	22.12	15.53	11.50	16.96	32.01
	B3. ASs2s	34.60	22.77	15.21	11.29	16.78	32.88
	B4. S2s-at-mp-gsa	35.36	22.38	15.88	11.85	17.63	33.02
	B5. S2s-at-mp-gsa (+cov, +ans)	38.74	24.89	17.88	13.48	18.39	34.51
	B6. CGC-QG	31.18	22.55	17.69	14.36	**25.20**	**40.94**
Proposed	P1. SRL-Graph	40.40	26.83	19.66	15.03	19.73	36.24
	P2. DP-Graph	**40.55**	**27.21**	**20.13**	**15.53**	20.15	36.94
Ablation	A1. *-w/o* Contexts	36.48	20.56	12.89	8.46	15.43	30.86
	A2. *-w/o* Semantic Graph	37.63	24.81	18.14	13.85	19.24	34.93
	A3. *-w/o* Multi-Relation & Attention	38.50	25.37	18.54	14.15	19.15	35.12
	A4. *-w/o* Multi-Task	39.43	26.10	19.14	14.66	19.25	35.76

Table 1: Performance comparison with baselines and the ablation study. The best performance is in bold.

Model	Short Contexts			Medium Contexts			Long Contexts			Average		
	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.
B4. S2sa-at-mp-gsa	3.76	4.25	3.98	3.43	4.35	4.13	3.17	3.86	3.57	3.45	4.15	3.89
B6. CGC-QG	3.91	4.43	3.60	3.63	4.17	4.10	**3.69**	3.85	**4.13**	3.75	4.15	3.94
A2. *-w/o* Semantic Graph	4.01	4.43	4.15	3.65	4.41	4.12	3.54	3.88	3.55	3.73	4.24	3.94
A4. *-w/o* Multi-Task	4.11	4.58	4.28	3.81	4.27	**4.38**	3.44	3.91	3.84	3.79	4.25	4.17
P2. DP-Graph	**4.34**	**4.64**	**4.33**	**3.83**	**4.51**	4.28	3.55	**4.08**	4.04	**3.91**	**4.41**	**4.22**
G1. Ground Truth	4.75	4.87	4.74	4.65	4.73	4.73	4.46	4.61	4.55	4.62	4.74	4.67

Table 2: Human evaluation results for different methods on inputs with different lengths. *Flu.*, *Rel.*, and *Cpx.* denote the *Fluency*, *Relevance*, and *Complexity*, respectively. Each metric is rated on a 1–5 scale (5 for the best).

3. While both SRL-based and DP-based semantic graph models (P1 and P2) achieve state-of-the-art BLEU, DP-based graph (P2) performs slightly better ($+3.3\%$ in BLEU-4). A possible explanation is that SRL fails to include fine-grained semantic information into the graph, as the parsing often results in nodes containing a long sequence of tokens.

4.4 Ablation Study

We also perform ablation studies to assess the impact of different components on the model performance against our DP-based semantic graph (P2) model. These are shown as Rows A1–4 in Table 1. Similar results are observed for the SRL-version.

• **Impact of semantic graph.** When we do not employ the semantic graph (A2, *-w/o* Semantic Graph), the BLEU-4 score of our model dramatically drops to 13.85, which indicates the necessity of building semantic graphs to model semantic relations between relevant content for deep QG. Despite its vital role, result of A1 shows that generating questions purely from the semantic graph is unsatisfactory. We posit three reasons: 1) the semantic graph alone is insufficient to convey the meaning of the entire document, 2) sequential information in the passage is not captured by the graph, and that 3) the automatically built semantic graph inevitably contains much noise. These reasons necessitate the composite document representation.

• **Impact of Att-GGNN.** Using a normal GGNN (A3, *-w/o* Multi-Relation & Attention) to encode the semantic graph, performance drops to 14.15 (-3.61%) in BLEU-4 compared to the model with Att-GGNN (A4, *-w/o* Multi-Task). This reveals that different entity types and their semantic relations provide auxiliary information needed to generate meaningful questions. Our Att-GGNN model (P2) incorporates attention into the normal GGNN, effectively leverages the information across multiple node and edge types.

• **Impact of joint training.** By turning off the content selection task (A4, *-w/o* Multi-Task), the BLEU-4 score drops from 15.53 to 14.66, showing the contribution of joint training with the auxiliary task of content selection. We further show that content selection helps to learn a QG-aware graph representation in Section 4.7, which trains the model to focus on the question-worthy content and form a correct reasoning chain in question decoding.

4.5 Human Evaluation

We conduct human evaluation on 300 random test samples consisting of: 100 short ($<$50 tokens), 100 medium (50-200 tokens), and 100 long ($>$200 tokens) documents. We ask three workers to rate the 300 generated questions as well as the ground-truth

Types		Examples	S2sa-at-mp-gsa	CGC-QG	DP-Graph
Correct	(Pred.) (G.T.)	Between Kemess Mine and Colomac Mine, which mine was operated earlier? What mine was operated at an earlier date, Kemess Mine or Colomac Mine?	56.5%	52.9%	67.4%
Semantic Error	(Pred.) (G.T.)	Lawrence Ferlinghetti is an American poet, **he is** a short story **written by who**? Lawrence Ferlinghetti is an American poet, he wrote a short story named what ?	17.7%	26.4%	8.3%
Answer Revealing	(Pred.) (G.T.)	What is the release date of this game **released on 17 October 2006**? What is the release date of this game named Hurricane?	2.1%	5.7%	1.4%
Ghost Entity	(Pred.) (G.T.)	When was the video game on which **Michael Gelling** plays Dr. Promoter? When was the video game on which Drew Gelling plays Dr. Promoter?	6.8%	0.7%	4.9%
Redundant	(Pred.) (G.T.)	What town did **Walcha and Walcha** belong to? What town did Walcha belong to?	16.3%	14.3%	13.9%
Unanswerable	(Pred.) (G.T.)	What is **the population of** the city Barack Obama was born? What was the ranking of the population of the city Barack Obama was born in 1999?	8.2%	18.6%	8.3%

Table 3: Error analysis on 3 different methods, with respects to 5 major error types (excluding the "Correct"). **Pred.** and **G.T.** show the example of the predicted question and the ground-truth question, respectively. **Semantic Error**: the question has logic or commonsense error; **Answer Revealing**: the question reveals the answer; **Ghost Entity**: the question refers to entities that do not occur in the document; **Redundant**: the question contains unnecessary repetition; **Unanswerable**: the question does not have the above errors but cannot be answered by the document.

questions between 1 (poor) and 5 (good) on three criteria: (1) *Fluency*, which indicates whether the question follows the grammar and accords with the correct logic; (2) *Relevance*, which indicates whether the question is answerable and relevant to the passage; (3) *Complexity*, which indicates whether the question involves reasoning over multiple sentences from the document. We average the scores from raters on each question and report the performance over five top models from Table 1. Raters were unaware of the identity of the models in advance. Table 2 shows our human evaluation results, which further validate that our model generates questions of better quality than the baselines. Let us explain two observations in detail:

• Compared against B4 (S2sa-at-mp-gsa), improvements are more salient in terms of "Fluency" ($+13.33\%$) and "Complexity" ($+8.48\%$) than that of "Relevance" ($+6.27\%$). The reason is that the baseline produces more shallow questions (affecting complexity) or questions with semantic errors (affecting fluency). We observe similar results when removing the semantic graph (A2. - *w/o* Semantic Graph). These demonstrate that our model, by incorporating the semantic graph, produces questions with fewer semantic errors and utilizes more context.

• All metrics decrease in general when the input document becomes longer, with the most obvious drop in "Fluency". When input contexts is long, it becomes difficult for models to capture question-worthy points and conduct correct reasoning, leading to more semantic errors. Our model tries to alleviate this problem by introducing semantic graph and content selection, but question quality drops as noise increases in the semantic graph as the document becomes longer.

4.6 Error Analysis

In order to better understand the question generation quality, we manually check the sampled outputs, and list the 5 main error sources in Table 3. Among them, "Semantic Error", "Redundant", and "Unanswerable" are noticeable errors for all models. However, we find that baselines have more unreasonable subject–predicate–object collocations (semantic errors) than our model. Especially, CGC-QG (B6) has the largest semantic error rate of 26.4% among the three methods; it tends to copy irrelevant contents from the input document. Our model greatly reduces such semantic errors to 8.3%, as we explicitly model the semantic relations between entities by introducing typed semantic graphs. The other noticeable error type is "Unanswerable"; *i.e.*, the question is correct itself but cannot be answered by the passage. Again, CGC-QG remarkably produces more unanswerable questions than the other two models, and our model achieves comparable results with S2sa-at-mp-gsa (B4), likely due to the fact that answerability requires a deeper understanding of the document as well as commonsense knowledge. These issues cannot be fully addressed by incorporating semantic relations. Examples of questions generated by different models is shown in Figure 3.

4.7 Analysis of Content Selection

We introduced the content selection task to guide the model to *select relevant content* and *form proper reasoning chains* in the semantic graph. To quantitatively validate the relevant content selection, we calculate the alignment of node attention

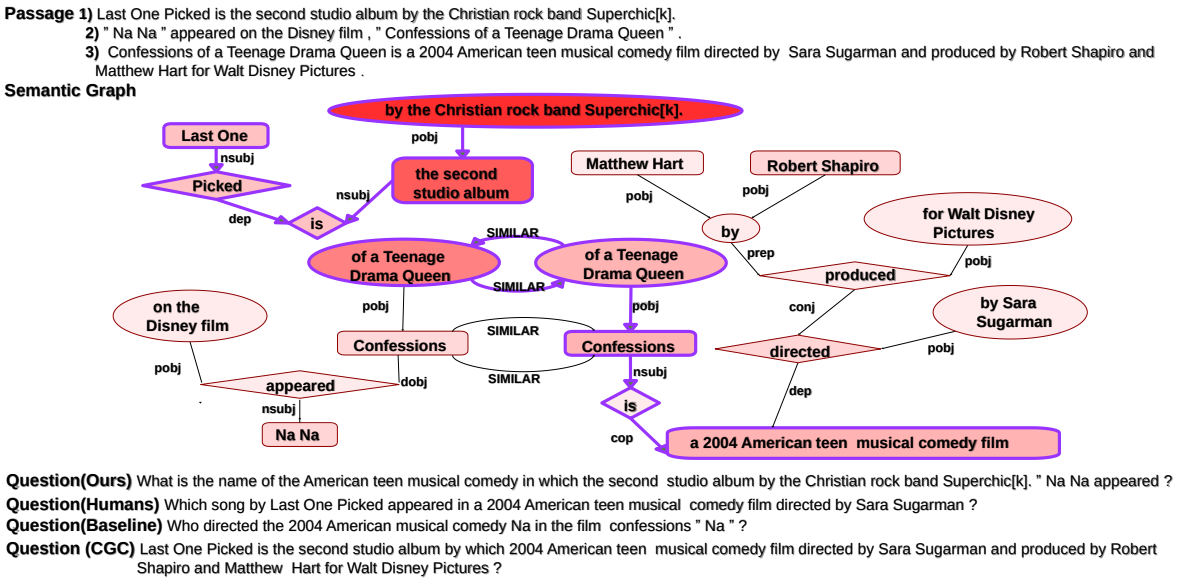


Figure 3: An example of generated questions and average attention distribution on the semantic graph, with nodes colored darker for more attention (best viewed in color).

α_{v_i} *w.r.t.* the relevant nodes $\sum_{v_i \in RN} \alpha_{v_i}$ and irrelevant nodes $\sum_{v_i \notin RN} \alpha_{v_i}$, respectively, under the conditions of both single training and joint training, where RN represents the ground-truth we set for content selection. Ideally, a successful model should focus on relevant nodes and ignore irrelevant ones; this is reflected by the ratio between $\sum_{v_i \in RN} \alpha_{v_i}$ and $\sum_{v_i \notin RN} \alpha_{v_i}$.

When jointly training with content selection, this ratio is 1.214 compared with 1.067 under single-task training, consistent with our intuition about content selection. Ideally, a successful model should concentrate on parts of the graph that help to form proper reasoning. To quantitatively validate this, we compare the concentration of attention in single- and multi-task settings by computing the entropy $H = -\sum \alpha_{v_i} \log \alpha_{v_i}$ of the attention distributions. We find that content selection increases the entropy from 3.51 to 3.57 on average. To gain better insight, in Figure 3, we visualize the semantic graph attention distribution of an example. We see that the model pays more attention (is darker) to the nodes that form the reasoning chain (the highlighted paths in purple), consistent with the quantitative analysis.

5 Conclusion and Future Works

We propose the problem of DQG to generate questions that requires reasoning over multiple disjoint pieces of information. To this end, we propose a novel framework which incorporates semantic graphs to enhance the input document representations and generate questions by jointly training with the task of content selection. Experiments on the HotpotQA dataset demonstrate that introducing semantic graph significantly reduces the semantic errors, and content selection benefits the selection and reasoning over disjoint relevant contents, leading to questions with better quality.

There are at least two potential future directions. First, graph structure that can accurately represent the semantic meaning of the document is crucial for our model. Although DP-based and SRL-based semantic parsing are widely used, more advanced semantic representations could also be explored, such as discourse structure representation (van Noord et al., 2018; Liu et al., 2019b) and knowledge graph-enhanced text representations (Cao et al., 2017; Yang et al., 2019). Second, our method can be improved by explicitly modeling the reasoning chains in generation of deep questions, inspired by related methods (Lin et al., 2018; Jiang and Bansal, 2019) in multi-hop question answering.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 2306–2317.

Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1623–1633.

Yllias Chali and Sadid A. Hasan. 2012. Towards automatic topical question generation. In *International Conference on Computational Linguistics (COLING)*, pages 475–492.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations (ICLR)*.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from wikipedia. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1907–1917.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1342–1352.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 866–874.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Michael Heilman. 2011. Automatic factual question generation from text. *Language Technologies Institute School of Computer Science Carnegie Mellon University*, 195.

Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4473–4483.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 6602–6609.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT@ACL)*, pages 228–231.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *International Conference on Learning Representations (ICLR)*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3243–3253.

Bang Liu, Mingjun Zhao, Di Niu, Kunfeng Lai, Yancheng He, Haojie Wei, and Yu Xu. 2019a. Learning to generate questions by learning what not to generate. In *International World Wide Web Conference (WWW)*, pages 1106–1118.

Jiangming Liu, Shay B. Cohen, and Mirella Lapata. 2019b. Discourse representation parsing for sentences and documents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6248–6262.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159.

Rik van Noord, Lasha Abzianidze, Antonio Toral, and Johan Bos. 2018. Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics (TACL)*, 6:619–633.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *CoRR*, abs/1905.08949.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4932–4942.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1073–1083.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3930–3939.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *CoRR*, abs/1710.10903.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. 2019. Auto-encoding scene graphs for image captioning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10685–10694.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380.

Xingdi Yuan, Tong Wang, Çaglar Gülçehre, Alessandro Sordoni, Philip Bachman, Saizheng Zhang, Sandeep Subramanian, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. In *The 2nd Workshop on Representation Learning for NLP (Rep4NLP@ACL)*, pages 15–25.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3901–3910.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *CCF International Conference of Natural Language Processing and Chinese Computing (NLPCC)*, pages 662–671.

A Supplemental Material

Here we give a more detailed description for the semantic graph construction, where we have employed two methods: *Semantic Role Labelling (SRL)* and *Dependency Parsing (DP)*.

A.1 SRL-based Semantic Graph

The primary task of semantic role labeling (SRL) is to indicate exactly what semantic relations hold among a predicate and its associated participants and properties (Màrquez et al., 2008). Given a document $\mathcal{D}$ with n sentences $\{s_1, \cdots, s_n\}$, Algorithm 1 gives the detailed procedure of constructing the semantic graph based on SRL.

Algorithm 1 Build SRL-based Semantic Graphs

Input: Document $\mathcal{D} = \{s_1, \cdots, s_n\}$
Output: Semantic graph $\mathcal{G}$

▷ build SRL graph
$\mathcal{D} \leftarrow$ COREFERENCE_RESOLUTION$(\mathcal{D})$
$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}, \mathcal{V} \leftarrow \emptyset, \mathcal{E} \leftarrow \emptyset$
for each sentence s **in** $\mathcal{D}$ **do**
  $\mathcal{S} \leftarrow$ SEMANTIC_ROLE_LABELING(s)
  for each tuple $\mathbf{t} = (a, v, m)$ **in** $\mathcal{S}$ **do**
    $\mathcal{V}, \mathcal{E} \leftarrow$ UPDATE_LINKS$(\mathbf{t}, \mathcal{V}, \mathcal{E})$
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{a, v, m\}$
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{\langle a, r^{a \to v}, v\rangle, \langle v, r^{v \to m}, m\rangle\}$
  end for
end for
▷ link to existing nodes
procedure UPDATE_LINKS$(\mathbf{t}, \mathcal{V}, \mathcal{E})$
  for each element e **in** $\mathbf{t}$ **do**
    for each node v_i **in** $\mathcal{V}$ **do**
      if IS_SIMILAR(v_i, e) **then**
        $\mathcal{E} \leftarrow \mathcal{E} \cup \{\langle e, r^s, v_i\rangle\}$
        $\mathcal{E} \leftarrow \mathcal{E} \cup \{\langle v_i, r^s, e\rangle\}$
      end if
    end for
  end for
end procedure
return $\mathcal{G}$

We first create an empty graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the node and edge sets, respectively. For each sentence s, we use the state-of-the-art BERT-based model (Shi and Lin, 2019) provided in the AllenNLP toolkit[3] to perform SRL, resulting a set of SRL tuples $\mathcal{S}$. Each tuple $\mathbf{t} \in \mathcal{S}$ consists of an argument a, a verb v, and (possibly) a modifier m, each of which is a text span of the sentence. We treat each of a, v, and m as a node and link it to an existing node $v_i \in \mathcal{V}$ if it is similar to v_i. Two nodes A and B are similar if one of following rules are satisfied: (1) A is equal to B; (2) A contains B; (3) the number of overlapped words between A and B is larger than the half of the minimum number of words in A and B. The edge between two similar nodes is associated with a special semantic relationship *SIMILAR*, denoted as r^s. Afterwards, we add two edges $\langle a, r^{a \to v}, v\rangle$ and $\langle v, r^{v \to m}, m\rangle$ into the edge set, where $r^{a \to v}$ and $r^{v \to m}$ denotes the semantic relationship between (a, v) and (v, w), respectively. As a result, we obtain a semantic graph with multiple node and edge types based on the SRL, which captures the core semantic relations between entities within the document.

[3] https://demo.allennlp.org/semantic-role-labeling

Algorithm 2 Build DP-based Semantic Graphs

Input: Document $\mathcal{D} = \{s_1, \cdots, s_n\}$
Output: Semantic graph $\mathcal{G}$

▷ Dependency parsing
$\mathcal{T} \leftarrow \emptyset$
$\mathcal{D} \leftarrow$ COREFERENCE_RESOLUTION$(\mathcal{D})$
for each sentence s **in** $\mathcal{D}$ **do**
  $T_s \leftarrow$ DEPENDENCY_PARSE(s)
  $T_s \leftarrow$ IDENTIFY_NODE_TYPES(T_s)
  $T_s \leftarrow$ PRUNE_TREE(T_s)
  $T_s \leftarrow$ MERGE_NODES(T_s)
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{T_s\}$
end for
▷ Initialize graph
$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}, \mathcal{V} \leftarrow \emptyset, \mathcal{E} \leftarrow \emptyset$
for each tree $T = (V_T, E_T)$ **in** $\mathcal{T}$ **do**
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{V_T\}$
  $\mathcal{E} \leftarrow \mathcal{E} \cup \{E_T\}$
end for
▷ Connect similar nodes
for each node v_i **in** $\mathcal{V}$ **do**
  for each node v_j **in** $\mathcal{V}$ **do**
    if $i \neq j$ **and** IS_SIMILAR(v_i, v_j) **then**
      $\mathcal{E} \leftarrow \mathcal{E} \cup \{\langle v_i, r^s, v_j\rangle, \langle v_j, r^s, v_i\rangle\}$
    end if
  end for
end for
return $\mathcal{G}$

A.2 DP-based Semantic Graph

Dependency Parsing (DP) analyzes the grammatical structure of a sentence, establishing relationships between "head" words and words that modify

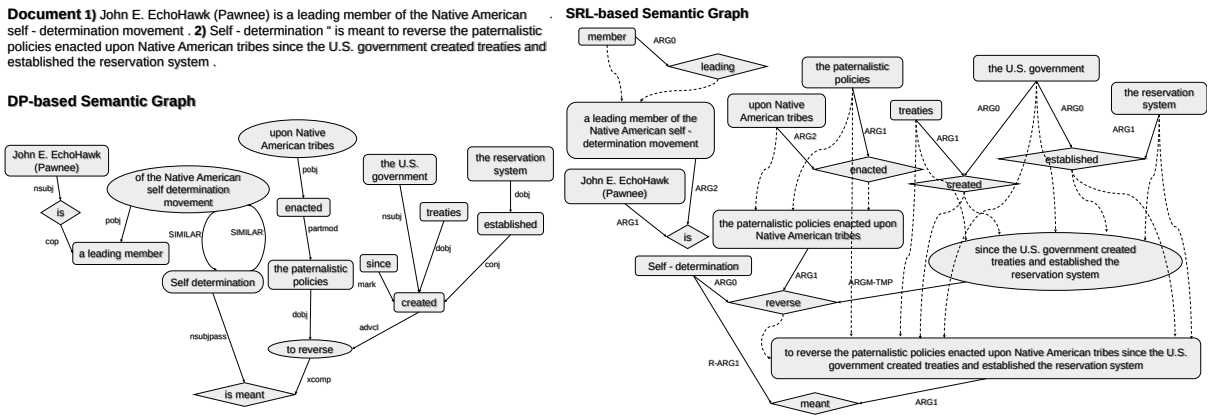


Figure 4: An example of constructed DP- and SRL- based semantic graphs, where $\dashrightarrow$ indicates *CHILD* relation, and rectangular, rhombic and circular nodes represent arguments, verbs and modifiers respectively.

them, in a tree structure. Given a document $\mathcal{D}$ with n sentences $\{s_1, \cdots, s_n\}$, Algorithm 2 gives the detailed procedure of constructing the semantic graph based on dependency parsing.

To better represent the entity connection within the document, we first employ the coreference resolution system of AllenNLP to replace the pronouns that refer to the same entity with its original entity name. For each sentence s, we employ the AllenNLP implementation of the biaffine attention model (Dozat and Manning, 2017) to obtain its dependency parse tree T_s. Afterwards, we perform the following operations to refine the tree:

• IDENTIFY_NODE_TYPES: each node in the dependency parse tree is a word associated with a POS tag. To simplify the node type system, we manually categorize the POS types into three groups: *verb*, *noun*, and *attribute*. Each node is then assigned to one group as its node type.

• PRUNE_TREE: we then prune each tree by removing unimportant continents (*e.g.*, punctuation) based on pre-defined grammar rules. Specifically, we do this recursively from top to bottom where for each node v, we visit each of its child node c. If c needs to be pruned, we delete c and directly link each child node of c to v.

• MERGE_NODES: each node in the tree represents only one word, which may lead to a large and noisy semantic graph especially for long documents. To ensure that the semantic graph only retains important semantic relations, we merge consecutive nodes that form a complete semantic unit. To be specific, we apply a simple yet effective rule: merging a node v with its child c if they form a consecutive modifier, *i.e.*, both the type of v and c are *modifier*, and v and c is consecutive in the sentence.

After obtaining the refined dependency parse tree T_s for each sentence s, we add intra-tree edges to construct the semantic graph by connecting the nodes that are similar but from different parse trees. For each possible node pair $\langle v_i, v_j \rangle$, we add an edge between them with a special edge type *SIMILAR* (denoted as r^s) if the two nodes are similar, *i.e.*, satisfying the same condition as described in Section A.1.

A.3 Examples

Figure 4 shows a real example for the DP- and SRL-based semantic graph, respectively. In general, DP-based graph contains less words for each node compared with the SRL-based graph, allowing it to include more fine-grained semantic relations. For example, *a leading member of the Native American self-determination movement* is treated as a single node in the SRL-based graph. While in the DP-based graph, it is represented as a semantic triple $\langle$ *a leading member*, *pobj*, *the Native American self-determination movement* $\rangle$. As the node is more fine-grained in the DP-based graph, this makes the graph typically more sparse than the SRL-based graph, which may hinder the message passing during graph propagation.

In experiments, we have compared the performance difference when using DP- and SRL-based graphs. We find that although both SRL- and DP-based semantic graph outperforms all baselines in terms of BLEU 1-4, DP-based graph performs slightly better than SRL-based graph ($+3.3\%$ in BLEU-4).