Natural Question Generation with Reinforcement Learning Based Graph-to-Sequence Model

Yu Chen
Rensselaer Polytechnic Institute
cheny39@rpi.edu

Lingfei Wu*
IBM Research
lwu@email.wm.edu

Mohammed J. Zaki
Rensselaer Polytechnic Institute
zaki@cs.rpi.edu

Abstract

Natural question generation (QG) aims to generate questions from a passage and an answer. In this paper, we propose a novel reinforcement learning (RL) based graph-to-sequence (Graph2Seq) model for QG. Our model consists of a Graph2Seq generator where a novel Bidirectional Gated Graph Neural Network is proposed to embed the passage, and a hybrid evaluator with a mixed objective combining both cross-entropy and RL losses to ensure the generation of syntactically and semantically valid text. The proposed model outperforms previous state-of-the-art methods by a large margin on the SQuAD dataset.

1   Introduction

Natural question generation (QG) is a dual task to question answering [4, 5].   Given a passage $X^p = \{x_1^p, x_2^p, ..., x_N^p\}$ and a target answer $X^a = \{x_1^a, x_2^a, ..., x_L^a\}$, the goal of QG is to generate the best question $\hat{Y} = \{y_1, y_2, ..., y_T\}$ which maximizes the conditional likelihood $\hat{Y} = \arg\max_Y P(Y|X^p, X^a)$. Recent works on QG mostly formulate it as a sequence-to-sequence (Seq2Seq) learning problem [8, 41, 15]. However, these methods fail to utilize the rich text structure that could complement the simple word sequence. Cross-entropy based sequence training has notorious limitations like exposure bias and inconsistency between train/test measurement [27, 37, 24]. To tackle these limitations, some recent QG approaches [30, 16] aim at directly optimizing evaluation metrics using Reinforcement Learning (RL) [36]. However, they generally do not consider joint mixed objective functions with both syntactic and semantic constraints for guiding text generation.

Early works on neural QG did not take into account the answer information when generating a question. Recent works [43, 12] have explored various means of utilizing the answers to make the generated questions more relevant. However, they neglect potential semantic relations between the passage and answer, and thus fail to explicitly model the global interactions among them.

To address these aforementioned issues, as shown in Fig. 1, we propose an RL based generator-evaluator architecture for QG, where the answer information is utilized by an effective Deep Alignment Network. Our generator extends Gated Graph Neural Networks [18] by considering both incoming and outgoing edge information via a Bidirectional Gated Graph Neural Network (BiGGNN) for encoding the passage graph, and then outputs a question using an RNN-based decoder. Our hybrid evaluator is trained by optimizing a mixed objective function combining both cross-entropy loss and RL loss. We also introduce an effective Deep Alignment Network for incorporating the answer information into the passage. The proposed model is end-to-end trainable, and outperforms previous state-of-the-art methods by a great margin on the SQuAD dataset.

2   An RL-based generator-evaluator architecture

Deep alignment network. Answer information is crucial for generating relevant and high quality questions from a passage. However, previous methods often neglect potential semantic relations
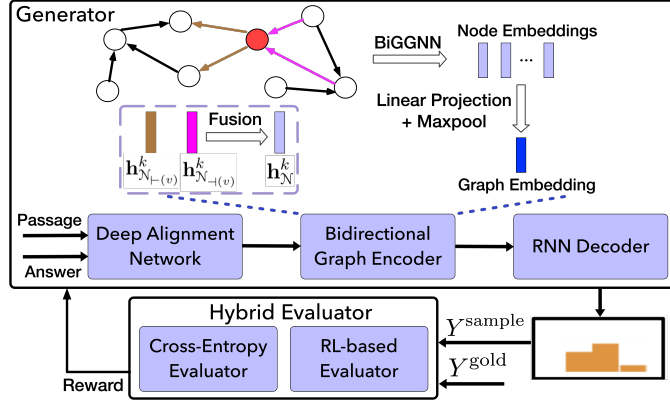
---

*Corresponding author.

Figure 1: Overall architecture of the proposed model. Best viewed in color.

between passage and answer words. We thus propose a novel Deep Alignment Network (DAN) for effectively incorporating answer information into the passage by performing soft-alignment at both word-level and contextualized hidden state level.

Let $\mathbf{X}^p \in \mathbb{R}^{F \times N}$ and $\widetilde{\mathbf{X}}^p \in \mathbb{R}^{\widetilde{F}_p \times N}$ denote two embeddings associated with passage text. Similarly, let $\mathbf{X}^a \in \mathbb{R}^{F \times L}$ and $\widetilde{\mathbf{X}}^a \in \mathbb{R}^{\widetilde{F}_a \times L}$ denoted two embeddings associated with answer text. Formally, we define our soft-alignment function as $\widetilde{\mathbf{H}}^p = \text{Align}(\mathbf{X}^p, \mathbf{X}^a, \widetilde{\mathbf{X}}^p, \widetilde{\mathbf{X}}^a) = \text{CAT}(\widetilde{\mathbf{X}}^p; \widetilde{\mathbf{X}}^a \boldsymbol{\beta}^T)$ where $\widetilde{\mathbf{H}}^p \in \mathbb{R}^{(\widetilde{F}_p + \widetilde{F}_a) \times N}$ is the final passage embedding, CAT denotes concatenation, and $\boldsymbol{\beta}$ is a $N \times L$ attention score matrix, computed by $\boldsymbol{\beta} \propto \exp(\text{ReLU}(\mathbf{W}\mathbf{X}^p)^T \text{ReLU}(\mathbf{W}\mathbf{X}^a))$. $\mathbf{W} \in \mathbb{R}^{d \times F}$ is a trainable weight matrix, with d being the hidden state size and ReLU is the rectified linear unit [22]. We next introduce how we do soft-alignment at both word-level and contextualized hidden state level.

Word-level alignment: In the word-level alignment stage, we first perform a soft-alignment between the passage and the answer based only on their pretrained GloVe embeddings and compute the final passage embeddings by $\widetilde{\mathbf{H}}^p = \text{Align}(\mathbf{G}^p, \mathbf{G}^a, [\mathbf{G}^p; \mathbf{B}^p; \mathbf{L}^p], \mathbf{G}^a)$, where $\mathbf{G}^p$, $\mathbf{B}^p$, and $\mathbf{L}^p$ are the corresponding GloVe embedding [25], BERT embedding [7], and linguistic feature (i.e., case, NER and POS) embedding of the passage text, respectively. Then a bidirectional LSTM [11] is applied to the passage embeddings $\widetilde{\mathbf{H}}^p = \{\widetilde{\mathbf{h}}_i^p\}_{i=1}^N$ to obtain contextualized passage embeddings $\bar{\mathbf{H}}^p \in \mathbb{R}^{\bar{F} \times N}$. Similarly, on the answer side, we simply concatenate its GloVe embedding $\mathbf{G}^a$ with its BERT embedding $\mathbf{B}^a$. Another BiLSTM is then applied to the above concatenated answer embedding sequence to obtain the contextualized answer embeddings $\bar{\mathbf{H}}^a \in \mathbb{R}^{\bar{F} \times L}$.

Hidden-level alignment: In the hidden-level alignment stage, we perform another soft-alignment based on the contextualized passage and answer embeddings. Similarly, we compute the aligned answer embedding, and concatenate it with the contextualized passage embedding to obtain the final passage embedding matrix $\text{Align}([\mathbf{G}^p; \mathbf{B}^p; \bar{\mathbf{H}}^p], [\mathbf{G}^a; \mathbf{B}^a; \bar{\mathbf{H}}^a], \bar{\mathbf{H}}^p, \bar{\mathbf{H}}^a)$. Finally, we apply another BiLSTM to the above concatenated embedding to get a $\bar{F} \times N$ passage embedding matrix $\mathbf{X}$.

Bidirectional graph encoder. Existing methods have exploited RNNs to capture local dependencies among sequence words, which, however, neglects rich hidden structured information in text. GNNs provide a better approach to utilize the rich text structure and to model the global interactions among sequence words [39, 40, 32]. Therefore, we explore various ways of constructing a passage graph $\mathcal{G}$ containing each word as a node, and then apply GNNs to encode the passage graph.

Passage graph construction: We explore syntax-based static graph construction. For each sentence in a passage, we first get its dependency parse tree. We then connect neighboring dependency parse trees by connecting those nodes that are at a sentence boundary and next to each other in text.

We also explore semantics-aware dynamic graph construction, which consists of three steps: i) we compute a dense adjacency matrix $\mathbf{A}$ for the passage graph by applying self-attention to the passage embeddings $\widetilde{\mathbf{H}}^p$; ii) KNN-style graph sparsification is adopted to obtain a sparse adjacency matrix $\bar{\mathbf{A}}$; and iii) we apply softmax to $\bar{\mathbf{A}}$ to get two normalized adjacency matrices, namely, $\mathbf{A}^{\dashv}$ and $\mathbf{A}^{\vdash}$, for incoming and outgoing directions, respectively. Please refer to Appendix A for more details.

Bidirectional gated graph neural networks: Unlike [38], we propose a novel BiGGNN which is an extension to Gated Graph Neural Networks (GGNNs) [18], to process the directed passage graph. Node embeddings are initialized to the passage embeddings $\mathbf{X}$ returned by DAN.

At each hop of BiGGNN, for every node, we apply an aggregation function which takes as input a set of incoming (or outgoing) neighboring node vectors and outputs a backward (or forward) aggregation vector, denoted as $\mathbf{h}^k_{\mathcal{N}_{\dashv(v)}}$ (or $\mathbf{h}^k_{\mathcal{N}_{\vdash(v)}}$). For the syntax-based static graph, the backward (or forward) aggregation vector is computed as the average of all incoming (or outgoing) neighboring node vectors plus itself. For the semantics-aware dynamic graph, we compute a weighted average for aggregation where the weights come from the adjacency matrices $\mathbf{A}^{\dashv}$ and $\mathbf{A}^{\vdash}$.

Unlike [38] that learns separate node embeddings for both directions independently, we fuse the information aggregated in two directions at each hop, defined as, $\mathbf{h}^k_{\mathcal{N}} = \text{Fuse}(\mathbf{h}^k_{\mathcal{N}_{\dashv(v)}}, \mathbf{h}^k_{\mathcal{N}_{\vdash(v)}})$. And the fusion function is designed as a gated sum of two information sources, defined as, $\text{Fuse}(\mathbf{a}, \mathbf{b}) = \mathbf{z} * \mathbf{a} + (1 - \mathbf{z}) * \mathbf{b}$ where $\mathbf{z} = \sigma(\mathbf{W}_z[\mathbf{a}; \mathbf{b}; \mathbf{a} * \mathbf{b}; \mathbf{a} - \mathbf{b}] + \mathbf{b}_z)$ with σ being a sigmoid function and $\mathbf{z}$ being a gating vector. Finally, a Gated Recurrent Unit (GRU) [6] is used to update the node embeddings by incorporating the aggregation information, defined as, $\mathbf{h}^k_v = \text{GRU}(\mathbf{h}^{k-1}_v, \mathbf{h}^k_{\mathcal{N}})$.

After n hops of GNN computation, where n is a hyperparameter, we obtain the final state embedding $\mathbf{h}^n_v$ for node v. To compute the graph-level embedding, we first apply a linear projection to the node embeddings, and then apply max-pooling over all node embeddings to get a d-dim vector $\mathbf{h}^{\mathcal{G}}$. The decoder takes the graph-level embedding $\mathbf{h}^{\mathcal{G}}$ followed by two separate fully-connected layers as initial hidden states (i.e., $\mathbf{c}_0$ and $\mathbf{s}_0$) and the node embeddings $\{\mathbf{h}^n_v, \forall v \in \mathcal{G}\}$ as the attention memory. Our decoder closely follows [29]. We refer the readers to Appendix B for more details.

Hybrid evaluator. Some recent QG approaches [30, 16] directly optimize evaluation metrics using REINFORCE to overcome the loss mismatch issue with cross-entropy based sequence training. However, they often fail to generate semantically meaningful and syntactically coherent text. To address these issues, we present a hybrid evaluator with a mixed objective combining both cross-entropy and RL losses so as to ensure the generation of syntactically and semantically valid text.

For the RL part, we adopt the self-critical sequence training (SCST) algorithm [28] to directly optimize evaluation metrics. In SCST, at each training iteration, the model generates two output sequences: the sampled output Y^s produced by multinomial sampling, and the baseline output $\hat{Y}$ obtained by greedy search. We define $r(Y)$ as the reward of an output sequence Y, computed by comparing it to corresponding ground-truth sequence Y^* with some reward metrics. The loss function is defined as $\mathcal{L}_{rl} = (r(\hat{Y}) - r(Y^s)) \sum_t \log P(y^s_t | X, y^s_{<t})$. As we can see, if the sampled output has a higher reward than the baseline one, we maximize its likelihood, and vice versa.

We use one of our evaluation metrics, BLEU-4, as our reward function f_{eval}, which lets us directly optimize the model towards the evaluation metrics. One drawback of some evaluation metrics like BLEU is that they do not measure meaning, but only reward systems for n-grams that have exact matches in the reference system. To make our reward function more effective and robust, following [9], we additionally use word mover's distance (WMD) [17] as a semantic reward function f_{sem}. We define the final reward function as $r(Y) = f_{\text{eval}}(Y, Y^*) + \alpha f_{\text{sem}}(Y, Y^*)$ where α is a scalar.

We train our model in two stages. In the first state, we train the model using regular cross-entropy loss. And in the second stage, we fine-tune the model by optimizing a mixed objective function combining both cross-entropy loss and RL loss. During the testing phase, we use beam search to generate final predictions. Further details of the training strategy can be found in Appendix C.

3 Experiments

In this section, we evaluate our proposed model against state-of-the-art methods on the SQuAD dataset [26]. The baseline methods in our experiments include SeqCopyNet [44], NQG++ [43], MPQG+R [30], Answer-focused Position-aware model [33], s2sa-at-mp-gsa [42], ASs2s [12] and CGC-QG [20]. Note that experiments on baselines followed by * are conducted using released source code. Detailed description of the baselines is provided in Appendix D. For fair comparison with baselines, we do experiments on both SQuAD split-1 [31] and split-2 [43]. For model settings and sensitivity analysis of hyperparameters, please refer to Appendix E and Appendix F. The implementation of our model will be made publicly available at `https://github.com/hugochan/RL-based-Graph2Seq-for-NQG`.

3

Following previous works, we use BLEU-4 [23], METEOR [2] and ROUGE-L [19] as our evaluation metrics. Note that we only report BLEU-4 on split-2 since most of the baselines only report this result. Besides automatic evaluation metrics, we also conduct human evaluation on split-2. Further details on human evaluation can be found in Appendix G.

Table 1: Automatic evaluation results on the SQuAD test set.

| Methods | Split-1 | | | Split-2 |
	BLEU-4	METEOR	ROUGE-L	BLEU-4
SeqCopyNet	–	–	–	13.02
NQG++	–	–	–	13.29
MPQG+R*	14.39	18.99	42.46	14.71
Answer-focused Position-aware model	–	–	–	15.64
s2sa-at-mp-gsa	15.32	19.29	43.91	15.82
ASs2s	16.20	19.92	43.96	16.17
CGC-QG	–	–	–	17.55
$G2S_{dyn}$+BERT+RL	17.55	21.42	45.59	18.06
$G2S_{sta}$+BERT+RL	**17.94**	**21.76**	**46.02**	**18.30**

Experimental results and analysis. Table 1 shows the automatic evaluation results comparing against all baselines. First of all, our full model $G2S_{sta}$+BERT+RL outperforms previous state-of-the-art methods by a great margin. Compared to previous methods like CGC-QG [20] and ASs2s [12] that rely on many heuristic rules and ad-hoc strategies, our model does not rely on any of these hand-crafted rules and ad-hoc strategies.

Table 2: Ablation study on the SQuAD split-2 test set.

Methods	BLEU-4	Methods	BLEU-4
$G2S_{dyn}$+BERT+RL	18.06	$G2S_{dyn}$	16.81
$G2S_{sta}$+BERT+RL	18.30	$G2S_{sta}$	16.96
$G2S_{sta}$+BERT-fixed+RL	18.20	$G2S_{dyn}$ w/o DAN	12.58
$G2S_{dyn}$+BERT	17.56	$G2S_{sta}$ w/o DAN	12.62
$G2S_{sta}$+BERT	18.02	$G2S_{sta}$ w/o BiGGNN, w/ Seq2Seq	16.14
$G2S_{sta}$+BERT-fixed	17.86	$G2S_{sta}$ w/o BiGGNN, w/ GCN	14.47
$G2S_{dyn}$+RL	17.18	$G2S_{sta}$ w/ GGNN-forward	16.53
$G2S_{sta}$+RL	17.49	$G2S_{sta}$ w/ GGNN-backward	16.75

We also perform ablation study on the impact of different components on the SQuAD split-2 test set, as shown in Table 2. For complete results on ablation study, please refer to Appendix H. By turning off DAN, the BLEU-4 score of $G2S_{sta}$ (similarly for $G2S_{dyn}$) dramatically drops from 16.96% to 12.62%, which shows the effectiveness of DAN. We can see the advantages of Graph2Seq learning over Seq2Seq learning by comparing the performance between $G2S_{sta}$ and Seq2Seq. Fine-tuning the model using REINFORCE can further improve the model performance, which shows the benefits of directly optimizing evaluation metrics. We also find that BERT has a considerable impact on the performance. Lastly, we find that static graph construction slightly outperforms dynamic graph construction. We refer the readers to Appendix I for a case study of different ablated systems.

4 Conclusion

We proposed a novel RL based Graph2Seq model for QG, where the answer information is utilized by an effective Deep Alignment Network and a novel bidirectional GNN is proposed to process the directed passage graph. Our two-stage training strategy benefits from both cross-entropy based and REINFORCE based sequence training. We also explore both static and dynamic approaches for constructing graphs when applying GNNs to textual data. On the SQuAD dataset, our model outperforms previous state-of-the-art methods by a wide margin. In the future, we would like to investigate more effective ways of automatically learning graph structures from free-form text.

References

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[2] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[3] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[4] Y. Chen, L. Wu, and M. J. Zaki. Bidirectional attentive memory networks for question answering over knowledge bases. *arXiv preprint arXiv:1903.02188*, 2019.

[5] Y. Chen, L. Wu, and M. J. Zaki. Graphflow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. *arXiv preprint arXiv:1908.00059*, 2019.

[6] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

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

[8] X. Du, J. Shao, and C. Cardie. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*, 2017.

[9] H. Gong, S. Bhat, L. Wu, J. Xiong, and W.-m. Hwu. Reinforcement learning based text style transfer without parallel training corpus. *arXiv preprint arXiv:1903.10671*, 2019.

[10] J. Gu, Z. Lu, H. Li, and V. O. Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] Y. Kim, H. Lee, J. Shin, and K. Jung. Improving neural question generation using answer separation. *arXiv preprint arXiv:1809.02393*, 2018.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

[15] V. Kumar, K. Boorla, Y. Meena, G. Ramakrishnan, and Y.-F. Li. Automating reading comprehension by generating question and answer pairs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 335–348. Springer, 2018.

[16] V. Kumar, G. Ramakrishnan, and Y.-F. Li. A framework for automatic question generation from text using deep reinforcement learning. *arXiv preprint arXiv:1808.04961*, 2018.

[17] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. In *International Conference on Machine Learning*, pages 957–966, 2015.

[18] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[19] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[20] B. Liu, M. Zhao, D. Niu, K. Lai, Y. He, H. Wei, and Y. Xu. Learning to generate questions by learning what not to generate. *arXiv preprint arXiv:1902.10418*, 2019.

[21] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[22] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[23] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[24] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[25] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[26] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[27] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[28] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024, 2017.

[29] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

[30] L. Song, Z. Wang, and W. Hamza. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058*, 2017.

[31] L. Song, Z. Wang, W. Hamza, Y. Zhang, and D. Gildea. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, 2018.

[32] A. Subburathinam, D. Lu, H. Ji, J. May, S.-F. Chang, A. Sil, and C. Voss. Neural question generation from text: A preliminary study. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[33] X. Sun, J. Liu, Y. Lyu, W. He, Y. Ma, and S. Wang. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, 2018.

[34] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, 2016.

[35] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[36] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[37] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[38] K. Xu, L. Wu, Z. Wang, and V. Sheinin. Graph2seq: Graph to sequence learning with attention-based neural networks. *arXiv preprint arXiv:1804.00823*, 2018.

[39] K. Xu, L. Wu, Z. Wang, M. Yu, L. Chen, and V. Sheinin. Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. *arXiv preprint arXiv:1808.07624*, 2018.

[40] K. Xu, L. Wu, Z. Wang, M. Yu, L. Chen, and V. Sheinin. Sql-to-text generation with graph-to-sequence model. *arXiv preprint arXiv:1809.05255*, 2018.

[41] K. Yao, L. Zhang, T. Luo, L. Tao, and Y. Wu. Teaching machines to ask questions.

[42] Y. Zhao, X. Ni, Y. Ding, and Q. Ke. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, 2018.

[43] Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer, 2017.

[44] Q. Zhou, N. Yang, F. Wei, and M. Zhou. Sequential copying networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

A   Details on dynamic graph construction

We dynamically build a directed and weighted graph to model semantic relationships among passage words. We make the process of building such a graph depend on not only the passage, but also on the answer. The graph construction procedure consists of three steps: i) we compute a dense adjacency matrix $\mathbf{A}$ for the passage graph by applying self-attention to the word-level passage embeddings $\widetilde{\mathbf{H}}^p$; ii) a KNN-style graph sparsification strategy is adopted to obtain a sparse adjacency matrix $\bar{\mathbf{A}}$, where we only keep the K nearest neighbors (including itself) as well as the associated attention scores (i.e., the remaining attentions scores are masked off) for each node; and iii) we apply softmax to $\bar{\mathbf{A}}$ to get two normalized adjacency matrices, namely, $\mathbf{A}^{\dashv}$ and $\mathbf{A}^{\vdash}$, for incoming and outgoing directions, respectively.

$$\mathbf{A} = \mathrm{ReLU}(\mathbf{U}\widetilde{\mathbf{H}}^p)^T \, \mathrm{ReLU}(\mathbf{U}\widetilde{\mathbf{H}}^p) \quad \bar{\mathbf{A}} = \mathrm{KNN}(\mathbf{A}) \quad \mathbf{A}^{\dashv}, \mathbf{A}^{\vdash} = \mathrm{softmax}(\{\bar{\mathbf{A}}, \bar{\mathbf{A}}^T\}) \quad (1)$$

where $\mathbf{U}$ is a $d \times (\widetilde{F}_p + \widetilde{F}_a)$ trainable weight matrix. Note that the supervision signal is able to back-propagate through the KNN-style graph sparsification operation since the K nearest attention scores are kept.

B   Details on the RNN decoder

On the decoder side, we adopt an attention-based [1, 21] LSTM decoder with copy [35, 10] and coverage mechanisms [34]. At each decoding step t, an attention mechanism learns to attend to the most relevant words in the input sequence, and computes a context vector $\mathbf{h}_t^*$ based on the current decoding state $\mathbf{s}_t$, the current coverage vector $\mathbf{c}^t$ and the attention memory. In addition, the generation probability $p_{\mathrm{gen}} \in [0, 1]$ is calculated from the context vector $\mathbf{h}_t^*$, the decoder state $\mathbf{s}_t$ and the decoder input y_{t-1}. Next, p_{gen} is used as a soft switch to choose between generating a word from the vocabulary, or copying a word from the input sequence. We dynamically maintain an extended vocabulary which is the union of the usual vocabulary and all words appearing in a batch of source examples (i.e., passages and answers). Finally, in order to encourage the decoder to utilize the diverse components of the input sequence, a coverage mechanism is applied. At each step, we maintain a coverage vector $\mathbf{c}^t$, which is the sum of attention distributions over all previous decoder time steps. A coverage loss is also computed to penalize repeatedly attending to the same locations of the input sequence.

C   Training details

We train our model in two stages. In the first state, we train the model using regular cross-entropy loss, defined as, $\mathcal{L}_{lm} = \sum_t -\log P(y_t^*|X, y_{<t}^*) + \lambda \, \mathrm{covloss}_t$ where y_t^* is the word at the t-th position of the ground-truth output sequence and $\mathrm{covloss}_t$ is the coverage loss defined as $\sum_i min(a_i^t, c_i^t)$, with a_i^t being the i-th element of the attention vector over the input sequence at time step t. Scheduled teacher forcing [3] is adopted to alleviate the exposure bias problem. In the second stage, we fine-tune the model by optimizing a mixed objective function combining both cross-entropy loss and RL loss,

defined as, $\mathcal{L} = \gamma\mathcal{L}_{rl} + (1 - \gamma)\mathcal{L}_{lm}$ where γ is a scaling factor controling the trade-off between cross-entropy loss and RL loss. A similar mixed-objective learning function has been used by [37, 24] for machine translation and text summarization.

D   Details on baseline methods

SeqCopyNet [44] proposed an extension to the copy mechanism which learns to copy not only single words but also sequences from the input sentence.

NQG++ [43] proposed an attention-based Seq2Seq model equipped with copy mechanism and a feature-rich encoder to encode answer position, POS and NER tag information.

MPQG+R [30] proposed an RL-based Seq2Seq model with a multi-perspective matching encoder to incorporate answer information. Copy and coverage mechanisms are applied.

Answer-focused Position-aware model [33] consists of an answer-focused component which generates an interrogative word matching the answer type, and a position-aware component which is aware of the position of the context words when generating a question by modeling the relative distance between the context words and the answer.

s2sa-at-mp-gsa [42] proposed a model which contains a gated attention encoder and a maxout pointer decoder to tackle the challenges of processing long input sequences. For fair comparison, we report the results of the sentence-level version of their model to match with our settings.

ASs2s [12] proposed an answer-separated Seq2Seq model which treats the passage and the answer separately.

CGC-QG [20] proposed a multi-task learning framework to guide the model to learn the accurate boundaries between copying and generation.

E   Model settings

We keep and fix the 300-dim GloVe vectors for the most frequent 70,000 words in the training set. We compute the 1024-dim BERT embeddings on the fly for each word in text using a (trainable) weighted sum of all BERT layer outputs. The embedding sizes of case, POS and NER tags are set to 3, 12 and 8, respectively. We set the hidden state size of BiLSTM to 150 so that the concatenated state size for both directions is 300. The size of all other hidden layers is set to 300. We apply a variational dropout [14] rate of 0.4 after word embedding layers and 0.3 after RNN layers. We set the neighborhood size to 10 for dynamic graph construction. The number of GNN hops is set to 3. During training, in each epoch, we set the initial teacher forcing probability to 0.75 and exponentially increase it to $0.75 * 0.9999^i$ where i is the training step. We set α in the reward function to 0.1, γ in the mixed loss function to 0.99, and the coverage loss ratio λ to 0.4. We use Adam [13] as the optimizer and the learning rate is set to 0.001 in the pretraining stage and 0.00001 in the fine-tuning stage. We reduce the learning rate by a factor of 0.5 if the validation BLEU-4 score stops improving for three epochs. We stop the training when no improvement is seen for 10 epochs. We clip the gradient at length 10. The batch size is set to 60 and 50 on data split-1 and split-2, respectively. The beam search width is set to 5. All hyperparameters are tuned on the development set.

F   Sensitivity analysis of hyperparameters

We study the effect of the number of GNN hops on our model performance. We conduct experiments of the G2S$_{sta}$ model on the SQuAD split-2 data by varying the number of GNN hops. Fig. 2 shows that our model is not very sensitive to the number of GNN hops and can achieve reasonable good results with various number of hops.

G   Details on human evaluation

We conducted a small-scale (i.e., 50 random examples per system) human evaluation on the split-2 data. We asked 5 human evaluators to give feedback on the quality of questions generated by a set
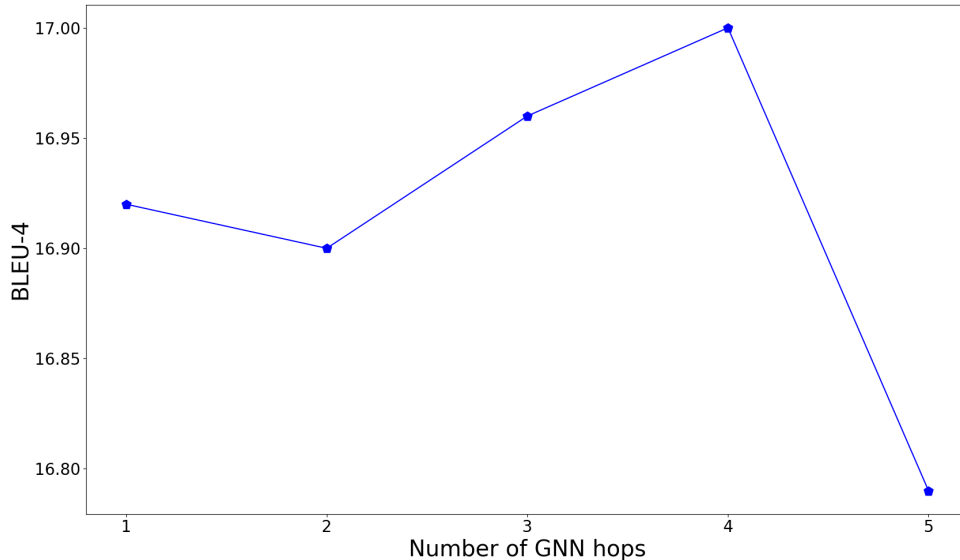
Figure 2: Effect of the number of GNN hops.

of anonymised competing systems. In each example, given a triple containing a source passage, a target answer and an anonymised system output, they were asked to rate the quality of the system output by answering the following three questions: i) is this generated question syntactically correct? ii) is this generated question semantically correct? and iii) is this generated question relevant to the passage? For each evaluation question, the rating scale is from 1 to 5, where a higher score means better quality (i.e., 1: Poor, 2: Marginal, 3: Acceptable, 4: Good, 5: Excellent). Responses from all evaluators were collected and averaged.

As shown in Table 3, we conducted a human evaluation study to assess the quality of the questions generated by our model, the baseline method MPQG+R, and the ground-truth data in terms of syntax, semantics and relevance metrics. We can see that our best performing model achieves good results even compared to the ground-truth, and outperforms the strong baseline method MPQG+R. Our error analysis shows that main syntactic error occurs in repeated/unknown words in generated questions. Further, the slightly lower quality on semantics also impacts the relevance.

Table 3: Human evaluation results on the SQuAD split-2 test set.

Methods	Syntactically correct (%)	Semantically correct (%)	Relevant (%)
MPQG+R*	4.34	4.01	3.21
$G2S_{sta}$+BERT+RL	4.41	4.31	3.79
Ground-truth	**4.74**	**4.74**	**4.25**

H Complete results on Ablation Study

We perform the comprehensive ablation study to systematically assess the impact of different model components (e.g., BERT, RL, DAN, BiGGNN, FEAT, DAN-word, and DAN-hidden) for two proposed full model variants (static vs dynamic) on the SQuAD split-2 test set. Our experimental results confirmed that every component in our proposed model makes the contribution to the overall performance.

I Case study of ablated systems

In Table 5, we further show a few examples that illustrate the quality of generated text given a passage under different ablated systems. As we can see, incorporating answer information helps the model identify the answer type of the question to be generated, and thus makes the generated questions

9

Table 4: Ablation study on the SQuAD split-2 test set.

Methods	BLEU-4	Methods	BLEU-4
G2S$_{dyn}$+BERT+RL	18.06	G2S$_{dyn}$ w/o feat	16.51
G2S$_{sta}$+BERT+RL	18.30	G2S$_{sta}$ w/o feat	16.65
G2S$_{sta}$+BERT-fixed+RL	18.20	G2S$_{dyn}$ w/o DAN	12.58
G2S$_{dyn}$+BERT	17.56	G2S$_{sta}$ w/o DAN	12.62
G2S$_{sta}$+BERT	18.02	G2S$_{sta}$ w/ DAN-word only	15.92
G2S$_{sta}$+BERT-fixed	17.86	G2S$_{sta}$ w/ DAN-hidden only	16.07
G2S$_{dyn}$+RL	17.18	G2S$_{sta}$ w/ GGNN-forward	16.53
G2S$_{sta}$+RL	17.49	G2S$_{sta}$ w/ GGNN-backward	16.75
G2S$_{dyn}$	16.81	G2S$_{sta}$ w/o BiGGNN, w/ Seq2Seq	16.14
G2S$_{sta}$	16.96	G2S$_{sta}$ w/o BiGGNN, w/ GCN	14.47

more relevant and specific. Also, we find our Graph2Seq model can generate more complete and valid questions compared to the Seq2Seq baseline. We think it is because a Graph2Seq model is able to exploit the rich text structure information better than a Seq2Seq model. Lastly, it shows that fine-tuning the model using REINFORCE can improve the quality of the generated questions.

Table 5: Generated questions on SQuAD split-2 test set. Target answers are underlined.

Passage: for the successful execution of a project , effective planning is essential .
Gold: what is essential for the successful execution of a project ?
G2S$_{sta}$ w/o BiGGNN (Seq2Seq): what type of planning is essential for the project ?
G2S$_{sta}$ w/o DAN.: what type of planning is essential for the successful execution of a project ?
G2S$_{sta}$: what is essential for the successful execution of a project ?
G2S$_{sta}$+BERT: what is essential for the successful execution of a project ?
G2S$_{sta}$+BERT+RL: what is essential for the successful execution of a project ?
G2S$_{dyn}$+BERT+RL: what is essential for the successful execution of a project ?
Passage: the church operates three hundred sixty schools and institutions overseas .
Gold: how many schools and institutions does the church operate overseas ?
G2S$_{sta}$ w/o BiGGNN (Seq2Seq): how many schools does the church have ?
G2S$_{sta}$ w/o DAN.: how many schools does the church have ?
G2S$_{sta}$: how many schools and institutions does the church have ?
G2S$_{sta}$+BERT: how many schools and institutions does the church have ?
G2S$_{sta}$+BERT+RL: how many schools and institutions does the church operate ?
G2S$_{dyn}$+BERT+RL: how many schools does the church operate ?