Vocabulary Matters: A Simple yet Effective Approach to Paragraph-level Question Generation

Vishwajeet Kumar
IITB-Monash Research Academy
Mumbai, India
vishwajeet@cse.iitb.ac.in

Manish Joshi
IIT Bombay
Mumbai, India
joshimanish0511@gmail.com

Ganesh Ramakrishnan
IIT Bombay
Mumbai, India
ganesh@cse.iitb.ac.in

Yuan-Fang Li
Monash University
Melbourne, Australia
yuanfang.li@monash.edu

Abstract

Question generation (QG) has recently attracted considerable attention. Most of the current neural models take as input only one or two sentences and perform poorly when multiple sentences or complete paragraphs are given as input. However, in real-world scenarios, it is very important to be able to generate high-quality questions from complete paragraphs. In this paper, we present a simple yet effective technique for answer-aware question generation from paragraphs. We augment a basic sequence-to-sequence QG model with dynamic, paragraph-specific dictionary and copy attention that is persistent across the corpus, without requiring features generated by sophisticated NLP pipelines or handcrafted rules. Our evaluation on SQuAD shows that our model significantly outperforms current state-of-the-art systems in question generation from paragraphs in both automatic and human evaluation. We achieve a 6-point improvement over the best system on BLEU-4, from 16.38 to 22.62.

1 Introduction and Related work

Automatic question generation (QG) from text aims to generate meaningful, relevant, and answerable questions from a given textual input. Owing to its applicability in conversational systems such as Cortana, Siri, chatbots, and automated tutoring systems, QG has attracted considerable interest in both academia and industry. Recent neural network-based approaches (Du et al., 2017; Kumar et al., 2018a,b; Du and Cardie, 2018; Zhao et al., 2018; Song et al., 2018; Subramanian et al., 2018; Tang et al., 2017; Wang et al., 2017) represent the state-of-the-art in question generation. Most of these techniques learn to generate questions from short text, *i.e.*, one or two sentences (Du et al., 2017; Kumar et al., 2018a,b; Du and Cardie, 2018). On the other hand, the ability to generate high-quality questions from longer text such as from multiple sentences or from a paragraph in its entirety, is more useful in real-world settings. However, given that a paragraph contains a longer context and more information than a sentence, it is a significantly more challenging problem to generate questions around a longer context. In figure 1 we present one motivating example demonstrating why the model needs information more than just a single sentence for generating question a meaningful and relevant question. As we can see in figure 1, question 2, question generated by our model use multiple sentences as context. Du et al. (2017) recently observed that 20% of the questions in the SQuAD dataset (Rajpurkar et al., 2016) require paragraph-level information to answer them. For the same reason, it is intuitive to conclude that the ability to consider the complete context; however long it may be, is critical for generating high-quality questions.

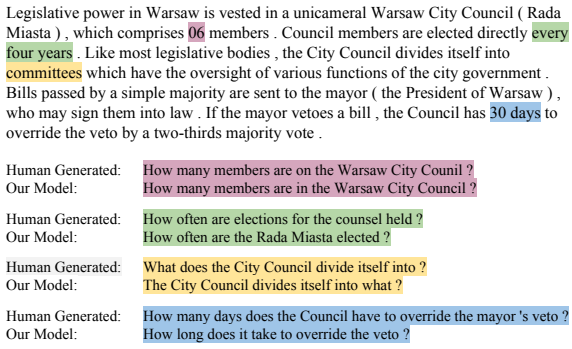

Figure 1: Examples of ground-truth questions and questions generated by our model from the same paragraph. Each question and its corresponding answer are highlighted using the same color.

Zhao et al. (2018) very recently proposed a technique (referred to MPGSN here) for paragraph-level question generation using a max out pointer mechanism and a gated self-attention encoder. Their best model achieves BLEU-4 of 16.38 on SQuAD with paragraphs as input. Compared to (Zhao et al.,

Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 781–785
December 4 - 7, 2020.

2018), our model has less number of parameters (making it more computationally efficient), is relatively easy to train and is somewhat deterministically biased toward the generation of important words in the input paragraph.

In this paper, we propose a simple yet effective paragraph-level question generation technique. We augment the standard sequence-to-sequence model based on bidirectional LSTM with two components: (1) a dynamic, paragraph-specific dictionary and (2) a copy attention mechanism that is persistent across paragraphs. Our evaluation on SQuAD shows significant improvement over MPGSN in automatic evaluation. We achieve a 6-point increase with respect to BLEU-4 (from 16.38 to 22.62) over MPGSN's best system. We perform the human evaluation of our model with and without copy attention, and we observe that we obtain 27% more relevant questions when the copy attention is incorporated.

For a given paragraph as input, we depict in Figure 1, the ground-truth questions as well as the questions generated along with the answers highlighted in the paragraph. As can be seen from the example, while generating the second question(highlighted in green color), our model uses information not only from the sentence containing the answer, but also relevant context from the complete paragraph.

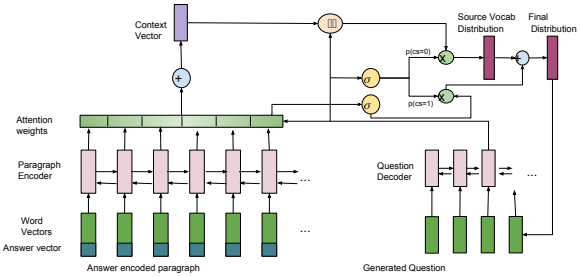

Figure 2: Overall architecture of our paragraph-level question generation model.

2 Problem Formulation & Approach

Given a paragraph 'P' and answer 'A', a question generation model iteratively samples question word $q_t \in V^Q$ at every time step 't' from the probability distribution given by:

$$\Pr(Q|P,A;\theta) = \prod_{t=1}^{|Q|} \Pr(q_t|P,A;\theta) \qquad (1)$$

Where V^Q is the question vocabulary, θ is the set of parameters, and A is the answer.

Our question generation model consists of a two-layer paragraph encoder and a one-layer question decoder, equipped with a dynamic dictionary and copy attention. In Figure 2, we illustrate the overall architecture of our paragraph level question generation model. The dynamic dictionary allows every training instance (paragraph) to have its own vocabulary instead of relying on the preprocessed global vocabulary. Copy attention enables the model to predict question words from the extended vocabulary (complete vocabulary + paragraph vocabulary). Copy attention operates over the union of words in vocabulary and paragraph words.

2.1 Paragraph encoder

We use a two-layer bidirectional long short-term memory (Bi-LSTM) network stack as the paragraph encoder. The paragraph encoder takes an answer-tagged paragraph as input and outputs a representation of the paragraph. Note that the Bi-LSTM network processes the input paragraph in both the forward and backward directions: $\overrightarrow{h_t} = LSTM(e_t, \overrightarrow{h_{t-1}})$ and $\overleftarrow{h_t} = LSTM(e_t, \overleftarrow{h_{t+1}})$, where $\overrightarrow{h_t}$ (resp. $\overleftarrow{h_t}$) is the forward (resp. backward) hidden state at time step t and e_t is the vector representation of current input x_t at time step t. The final hidden state for the current word input is the concatenation of the forward and backward hidden state vectors: $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$.

2.2 Dynamic, shared dictionary

In the traditional approach, a new/unknown word is typically replaced with the "<unk>" token. The copy mechanism (Gu et al., 2016) then unfortunately learns to copy this "<unk>" token instead of the actual (unknown) word from the source paragraph. Instead, we use a separate dynamic dictionary unique to each source paragraph, which includes all and only words that occur in the paragraph. This allows our model to copy source words that may not be in the target dictionary into the target (question). Using a dynamic dictionary consisting of the preprocessed vocabulary instead of a static one enables the copy mechanism to copy the exact words directly into the question, even if they are rare and unknown.

Given a source paragraph p, we denote its dynamic vocabulary by V^p. Our copy attention mechanism takes into account V^p and the global vocabulary V to determine whether to copy a word from V^p or to predict a word from question vocabulary V^Q.

As our model's source as well as target are in

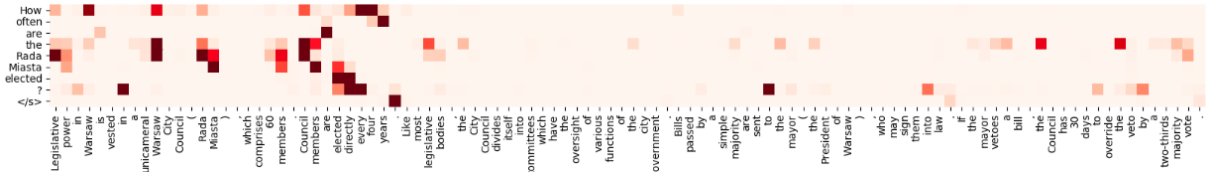

Figure 3: Visualizing attention weights for the second generated question in Fig. 1.

the same language, we work with a shared source and target vocabulary, though we learn different language models for the paragraph and the question. Sharing source and target vocabulary also decreases the memory requirement resulting from matrix multiplication (thus making faster training through larger batch size) possible. It also enables efficient question decoding, thus reducing the time for inference on the test data.

2.3 Question decoder

Our question decoder is another Bi-LSTM that takes as input the last hidden state and context representation from the encoder and generates question words sequentially based on the previously generated words. The decoder hidden state ($s_t = [\overrightarrow{s_t}, \overleftarrow{s_t}]$) at time step t is the concatenation of the forward and backward hidden state representations: $\overrightarrow{s_t} = LSTM(o_t, \overrightarrow{s_{t-1}})$ and $\overleftarrow{s_t} = LSTM(o_t, \overleftarrow{s_{t+1}})$, where o_t is the vector representation of decoder input (y_t) at time step t. During training time the vector representation of words from the ground-truth question is fed as decoder input, and during test time the vector representation of the vocabulary word with maximum probability is fed as input. We feed **EOS** symbol as input to decoder from both forward and backward dircetion at time t_0. Bidirectional decoder factorizes the conditional decoding probabilities in both directions (left-to-right and right-to-left) into summation as:

$$P\left(y_t|[y_m]_{m\neq t}\right) = \overrightarrow{\log p\left(y_t|Y_{[1:t-1]}\right)} + \overleftarrow{\log P\left(y_t|Y_{[t+1:T_y]}\right)} \quad (2)$$

The probability distribution over words in the vocabulary is calculated as:

$$\Pr(q_t) = softmax(\boldsymbol{W}_g \sigma(\boldsymbol{W}_s[s_t, h_t] + \boldsymbol{b}_s) + \boldsymbol{b}_g) \quad (3)$$

where $\boldsymbol{W}_g$, $\boldsymbol{W}_s$, $\boldsymbol{b}_s$ and $\boldsymbol{b}_g$ are trainable model parameters. Probability distribution $P(q_t)$ uses the standard softmax over the question vocabulary V^Q. This is used to sample word with maximum probability while decoding a question.

2.4 Copy attention

We know that a good question should be relevant to (answerable from) the paragraph. So we learn a probabilistic mixture model over the question vocabulary V^Q and the current paragraph vocabulary V^P. The current paragraph vocabulary is generated by a dynamic dictionary module.

Our copy attention calculates two values:

cs**:** a binary-valued variable which acts a switch between copying a word from the paragraph's dynamic vocabulary V^P or generating from the question vocabulary V^Q

$\Pr(.|V^P)$**:** probability of copying a particular word from paragraph vocabulary V^P.

Therefore, the final probability distribution from which a word will be sampled while generating a question is calculated over the extended vocabulary $V^Q \cup V^p$. Given a word from the extended vocabulary $w \in V^Q \cup V^P$, its probability $\Pr(w)$ is computed as:

$$\Pr(w) = \Pr(cs=1)\Pr\left(w|V^P\right) + \Pr(cs=0)\Pr\left(w|V^Q\right) \quad (4)$$

The switch probability $\Pr(cs)$ is determined using the decoder hidden states as:

$$\Pr(cs=1) = \sigma(\boldsymbol{W}_{cs} s_t + \boldsymbol{b}_{cs}) \quad (5)$$

where $\boldsymbol{W}_{cs}$ and $\boldsymbol{b}_{cs}$ are trainable model parameters. $\Pr\left(w|V^Q\right)$ is the probability of predicting a word from complete vocabulary V^Q. The copy attention weight a^t is computed as:

$$e_i^t = v^T tanh(\boldsymbol{W_h} h_i + \boldsymbol{W_s} s_t + b_{attn}) \quad (6)$$

$$a^t = sparsemax(e^t) \quad (7)$$

Where v, $\boldsymbol{W_h}$, $\boldsymbol{W}_s$ and $\boldsymbol{b}_{attn}$ are trainable model parameters. The probability of copying a word from the paragraph vocabulary V^P is estimated as:

$$\Pr\left(w|V^P\right) = \sigma(\boldsymbol{W}_a a^t + \boldsymbol{b}_a) \quad (8)$$

where $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ are trainable model parameters.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
MPGSN (Zhao et al., 2018)	45.07	29.58	21.60	16.38	20.25	44.48
L2A (Du et al., 2017)	42.54	25.33	16.98	11.86	16.28	39.37
NQG$_{dd}$ [w/o copy attention]	55.32	32.39	20.12	12.86	17.00	42.77
NQG$_{dd}$ [with copy attention]	**61.84**	**41.73**	**30.19**	**22.62**	**21.93**	**48.60**

Table 1: Results on the test set on automatic evaluation metrics. Best results for each metric (column) are **bolded**.

3 Experimental Setup

We report the experimental result of our model (referred to as NQG$_{dd}$) and compare it with the current state of the art MPGSN (Zhao et al., 2018). We employ the widely-used metrics BLEU (Papineni et al., 2002), ROUGE-L and METEOR for automatic evaluation. We use evaluation script provided by (Chen et al., 2015). Similar to (Kumar et al., 2018a) we also report qualitative assessment on the syntax, semantics and relevance of the questions generated by our model.

All experiments are performed on the SQuAD dataset (Rajpurkar et al., 2016), where complete paragraphs are taken as input instead of just one or two sentences. We reformat the SQuAD dataset such that during training time, each source instance is a (paragraph, question) pair annotated with the gold answers, and the target is a question. Following the exact setup from MPGSN (Zhao et al., 2018), we split the SQuAD train set into train and validation set containing 77,526 and 9,995 instances respectively, and take the separate SQuAD dev set containing 10,556 instances as our test set.

4 Results and Analysis

Table 1 summarizes results of the automatic evaluation of the test set. As can be seen, our model significantly outperforms the state-of-the-art MPGSN on all metrics. The improvements on BLEU are especially substantial, the BLEU-4 score of MPGSN is 16.38, and ours (with copy incorporated) is 22.62, an improvement of 6.24, or 38%. This large performance difference demonstrates the effectiveness of our dynamic dictionary.

In Table 2 we present human evaluation results. We evaluate the quality of questions generated in terms on *syntactic* correctness, *semantic* correctness and *relevance* to the paragraph. The evaluation is performed on a randomly selected subset of 100 sentences from the test set. Each of the three evaluators are presented the 100 paragraph-question pairs for two variants of our model (with and without copy) and asked for a binary responses for all three parameters. We averaged responses received by all three evaluators to compute the final scores. As can be seen, the incorporation of the copy attention improves performance, especially on relevance. We also measure the inter-rater agreement using Randolph's free-marginal multirater kappa (Randolph, 2005). It can be observed that our quality metrics for both our models are rated as *substantial agreement* (Viera et al., 2005).

To explain how our model attends to different words in the source paragraph we visualize attention weights in Figure 3, which shows attention weights between question 2 generated by our model and the corresponding paragraph in Figure 1. We observe that the attention weight is high for words near the answer and the model attends to all relevant context rather that just the sentence containing the answer.

Model	Syntax		Semantics		Relevance	
	Score	Kappa	Score	Kappa	Score	Kappa
NQG$_{dd}$ [w/o copy]	89	0.68	**83**	0.69	43	0.67
NQG$_{dd}$ [with copy]	**94**	0.64	82	0.68	**71**	0.73

Table 2: Human evaluation results (columns "Score") as well as inter-rater agreement (columns "Kappa") for each of our two models on 100 questions from the test set. The scores are between 0 (worst) and 100 (best). Best results for each metric (column) are in **bold**.

We also note that our training is faster atleast by a factor of 2. We expected this since we replace a slightly expensive self-attention mechanism in the decoder of (Zhao et al., 2018) with a simpler dynamic dictionary and reusable copy attention.

5 Conclusion

Paragraph-level question generation (QG) is an important but challenging problem, mainly due to the challenge in effectively handling a longer context. We present a simple yet effective approach for automatic question generation from paragraphs. Besides using a standard global source dictionary, our RNN-based model incorporates a dynamic, paragraph-specific dictionary, and learns to switch between copying from the combined

vocabulary and generating a new word. Through our experiments, we demonstrate how our model outperforms the current state-of-the-art model in paragraph-level QG by a wide margin, for example by 6.24 BLEU-4 points, a 38% improvement.

References

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *ACL (1)*, pages 1907–1917.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1342–1352.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.

Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018a. Automating reading comprehension by generating question and answer pairs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 335–348. Springer.

Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018b. A framework for automatic question generation from text using deep reinforcement learning. *arXiv preprint arXiv:1808.04961*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. 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.

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

Justus J Randolph. 2005. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. 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)*, volume 2, pages 569–574.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Anthony J Viera, Joanne M Garrett, et al. 2005. Understanding interobserver agreement: the kappa statistic. *Fam Med*, 37(5):360–363.

Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450*.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. 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.