ParaQG: A System for Generating Questions and Answers from Paragraphs

Vishwajeet Kumar1,3,4, Sivaanandh Munneeswaran2, Ganesh Ramakrishnan3, and Yuan-Fang Li4

1IITB-Monash Research Academy, Mumbai, India
2Mepco Schlenk Engineering College, Tamilnadu, India
3IIT Bombay, Mumbai, India
4Monash University, Melbourne, Australia

Abstract

Generating syntactically and semantically valid and relevant questions from paragraphs is useful with many applications. Manual generation is a labour-intensive task, as it requires the reading, parsing and understanding of long passages of text. A number of question generation models based on sequence-to-sequence techniques have recently been proposed. Most of them generate questions from sentences only, and none of them is publicly available as an easy-to-use service. In this paper, we demonstrate ParaQG, a Web-based system for generating questions from sentences and paragraphs. ParaQG incorporates a number of novel functionalities to make the question generation process user-friendly. It provides an interactive interface for a user to select answers with visual insights on generation of questions. It also employs various faceted views to group similar questions as well as filtering techniques to eliminate unanswerable questions.

1 Introduction

Asking relevant and intelligent questions has always been an integral part of human learning, as it can help assess user understanding of a piece of text (a comprehension, an article, etc.). However, forming questions manually is an arduous task. Automated question generation (QG) systems can help alleviate this problem by learning to generate questions on a large scale efficiently. A QG system has many applications in a wide variety of areas such as FAQ generation, intelligent tutoring systems, automating reading comprehension, and virtual assistants/chatbots. For a QG system, the task is to generate syntactically coherent, semantically correct and natural questions from text. Additionally, it is highly desirable that the questions are relevant to the text and are pivoted on answers present in the text. Distinct from other natural language generation tasks such as summarisation and paraphrasing, answers play an important role in question generation. Different questions can be formed from the same passage based on the choice of the pivotal answer. The pivotal answer is the span of text from the input passage around which a question is generated. The pivotal answer can be either selected manually by the user, automatically by the system or by a combination of the two. For example in Figure 1 it can be seen that based on different answers selected (highlighted in different colours), our system generates different questions.

Neural network-based sequence-to-sequence (Seq2Seq) models represent the state-of-the-art in question generation. Most of these models (Du et al., 2017; Kumar et al., 2018; Song et al., 2018; Kumar et al., 2019c,b) take single sentence as input, thus limiting their usefulness in real-world settings. Some recent techniques tackle the problem of question generation from paragraphs (Zhao et al., 2018). However, none of the above works is publicly available as an online service.

In this work we present ParaQG, an interactive Web-based question generation system to generate correct, meaningful and relevant questions.
from sentences, and paragraphs. Given a pas-
sage of text as input, users can manually select
a set of answer spans to ask questions about (i.e.
choose answers) from an automatically curated set
of noun phrases and named entities. Questions
are then generated by a combination of a (novel)
sequence-to-sequence model with dynamic dictio-
naries, the copy mechanism (Gu et al., 2016) and
the global sparse-max attention (Martins and As-
tudillo, 2016).

ParaQG incorporates the following main fea-
tures.

1. An interactive, user-configurable Web appli-
cation to automatically generate questions
from a sentence, or a paragraph based on user
selected answers, with visual insights on the
generated questions.

2. A technique to create faceted views of the
generated questions having overlapping or
similar answers. Given an input passage,
the same answer may appear multiple times
in different spans, from which similar ques-
tions can be generated. ParaQG detects and
presents the generated questions based on
a grouped/faceted view of similar answer
spans, thus allowing easy selection by users.

3. A novel question filtering technique based on
BERT (Devlin et al., 2018) to eliminate unan-
swerable questions from the text.

To the best of our knowledge we are the first to
propose and develop an interactive system that
generates questions based on the answers selected
by users. The rest of the paper is organized as fol-
lows. We discuss related work in Section 2. In
Section 3, We describe the architecture of ParaQG.
This is followed by details of the demonstration
in Section 4 and the implementation in Section 5.
Conclusion is discussed in Section 6.

2 Related Work

Automatically generating questions and answers
from text is a challenging task. This task can
be traced back to 1976 when Wolfe (1976) pre-
sented their system AUTOQUEST, which exam-
ined the generation of Wh-questions from single
sentences. This was followed by several pattern
matching (Hirschman et al., 1999) and linear re-
gression (Ng et al., 2000) based models. These
approaches are heavily dependent on either rules
or question templates, and require deep linguistic
knowledge, yet are not exhaustive enough. Recent
successes in neural machine translation (Sutskever
et al., 2014; Cho et al., 2014) have helped address
these issues by letting deep neural nets learn the
implicit rules from data. This approach has in-
spired application of sequence-to-sequence learn-
ing to automated question generation. Serban
et al. (2016) proposed an attention-based (Bah-
danau et al., 2014; Luong et al., 2015) approach
to question generation from a pre-defined template
of knowledge base triples (subject, relation, ob-
ject). We proposed multi-hop question generation
(Kumar et al., 2019a) from knowledge graphs us-
ing transformers (Vaswani et al., 2017). Du et
al. (2017) proposed an attention-based sequence
learning approach to question generation.

Most existing work focuses on generating ques-
tions from text without concerning itself with
answer generation. In our previous work (Ku-
mar et al., 2018), we presented a pointer network-
based model that predicts candidate answers and
generates a question by providing a pivotal answer
as an input to the decoder. Our model for question
generation combines a rich set of linguistic fea-
tures, pointer network-based answer selection, and
an improved decoder, and is able to generate ques-
tions that are relatively more relevant to the given
sentence than the questions generated without the
answer signal.

Overall, the broad finding has been that it is
important to either be provided with or learn to
choose pivotal answer spans to ask questions about
from an input passage. Founded on this observa-
tion, our system facilitates users with an option to
either choose answer spans from the pre-populated
set of named entities and noun phrases or manu-
ally select custom answer spans interactively. Our
system, ParaQG, presented in this paper uses a
novel four-stage procedure: (1) text review, (2)
pivotal answer selection (3) automatic question
generation pertaining to the selected answer, and
(4) filtering and grouping questions based on con-
fidence scores and different facets of the selected
answer.

3 System Architecture

ParaQG generates questions from sentences and
paragraphs following a four-stage interactive pro-
cedure: (a) paragraph review, (b) answer selec-
tion, (c) question generation with associated confi-
dence score, and (d) question filtering and grouping based on answer facets. Given a paragraph, ParaQG first reviews the content automatically and then flags any unprocessable characters (e.g. Unicode characters) and URLs, which the user are prompted to edit or remove (Section 3.1). Next, the user is provided with an option to select an answer from the list of candidate answers identified by the system. Alternatively, the user can select custom answer spans from the passage to ask question about (Section 3.2). In the third step, the selected pivotal spans are encoded into the paragraph and fed to the question generation module. The question generation module is a sequence-to-sequence model with dynamic dictionaries, reusable copy attention and global sparse-max attention. This module attempts to automatically generate the most relevant as well as syntactically and semantically correct questions around the selected pivotal answers (Section 3.3). In the last step the unanswerable questions are filtered out using a BERT-based question filtering module (Section 3.4). The questions that remain are presented by grouping their associated answers. Each group of answers (which we also refer to as an answer-based facet) corresponds to some unique stem form of those answer words.

3.1 Paragraph Review

Since every sentence/word in a paragraph may not be question-worthy, it is important to filter out those that are not. Given a paragraph text, the system automatically reviews its contents to check if the paragraph contains any non-ASCII characters, URLs etc., and flags them for users to edit, as illustrated in Figure 2a.

3.2 Answer Selection

ParaQG allows users to select any named entity or noun phrase present in the paragraph as a pivotal answer. As mentioned earlier, a user is presented with a list of all the named entities and noun phrases as extracted using the Stanford CoreNLP tagger to choose pivotal answers from. Alternatively, users can manually select a set of spans from the passage as pivotal answers, as shown in Figure 2d. The selected answer is encoded in the source sentence using the BIO (Begin, Inside, Outside) notation.

3.3 Question Generation

Similar to our previous work (Kumar et al., 2018), we encode the pivotal answer spans in the passage using BIO notation, and train a sequence-to-sequence model augmented with dynamic dictionary, copy mechanism and global sparse-max attention. Our question generation module consists of a paragraph encoder and a question decoder. The encoder represents the paragraph input as a single fixed-length continuous vector. This vector representation of the paragraph is passed to the decoder with reusable copy mechanism and sparse-max attention to generate questions.

3.4 BERT-based Question Filtering

We use the BERTbase (Devlin et al., 2018) model to filter out unanswerable questions generated by our model. We fine-tune BERT on SQuAD 2.0 (Rajpurkar et al., 2018). SQuAD 2.0 extends SQuAD with over 50000 unanswerable questions. The unanswerable questions are flagged with the attribute is_impossible.

We represent input question (question generated by our QG model) and the passage in a single packed sequence of tokens, while using a special token [SEP] to separate the question from the passage. Similar to (Devlin et al., 2018) we use a special classification token [CLS] at the start of every sequence. Let us denote the final hidden representation of the [CLS] token by $C$ and the final hidden representation for the $i^{th}$ input token by $T_i$. For each unanswerable question, we represent the start and end answer index using a [CLS] token as it does not have any answer start and end index. Similar to (Devlin et al., 2018), we compare the score of no-answer span with the score of best non-null answer span to predict the answerability of a question. Score of no-answer span is calculated as: $s_{null} = S \cdot C + E \cdot C$ where $S \in \mathbb{R}^H$ is the vector representation of answer start index and $E \in \mathbb{R}^H$ is the vector representation of answer end index. The score of a non-null answer span is defined as $s_{i,j} = \max_{j>i} \{S \cdot T_i + E \cdot T_j\}$ If the score of $s_{null} - s_{i,j} > V$, where $V$ is a threshold calculated using a validation set, then the question is not answerable using the paragraph.

3.5 Grouped/Faceted Views of Questions

We group together all answers and their corresponding question(s) that have the same stemmed form. For example, two potential answer spans
‘switching’ and ‘switches’ would have the same stemmed form ‘switch’. Thus, the spans ‘switching’ and ‘switches’, and their associated question(s) would be grouped together under the same stemmed form ‘switch’. Summarily, each such question group yields a faceted view of the question set. Within each group, the questions are sorted in decreasing order of their probabilities. We calculate the intra-question probability (confidence score) by normalizing the beam score $x$ as: $e^x / (1 + e^x)$. The final inter-question probability of a question-answer pair is calculated from the question with maximum intra-question probability $p$ as: $\frac{p - \min(P)}{\max(P) - \min(P)}$, where $P$ is the set of maximum probability scores across answers.

4 Demonstration Details

ParaQG is available as an interactive and fully-featured Web application. A video of the ParaQG system is available at https://youtu.be/BLChd18kz1c. The ParaQG system is accessible at https://www.cse.iitb.ac.in/~vishwajeet/paraqg.html. Important features of the Web application are discussed below.

Input and content review: A user can copy any paragraph and paste it in the text area (Fig. 2a), and subsequently will be asked to review and remove/edit unprocessable contents (Fig. 2a).

Interactive pivotal answer selection: ParaQG provides an interactive user interface for users to select pivotal answers. A user has an option to select a pivotal answers either from a set of noun phrases or from a set of named entities present in the paragraph. To choose a pivotal answer from a set of named entities, the user can click on the Named Entities tab (Figure 2b). Similarly, to select a noun phrase present in the paragraph as the pivotal answer, the user can click on the Noun Phrases tab (Figure 2c). Once a user clicks on either of the tabs he/she will be presented with pre-highlighted noun phrases/named entities as pivotal answers. The user can subsequently deselect a pivotal answer by clicking on it.

Custom pivotal answer selection: Alternatively, the user can click on the Custom Answers tab (Figure 2d) and manually select the most important spans in the paragraph as the pivotal answers. The users can also select overlapping spans.

Automatic question and answer generation: Finally, the user is presented with the question generated as well as the answer to that question along with confidence score. For example for the paragraph input by the user in Figure 2b, the questions as well as the answers are generated and shown to the user (Figure 3) along with their confidence score.

Visualization of decoder attention weights using heat maps: ParaQG also presents to the user heat maps of the decoder attention weights between words in the paragraph and words in the question generated. A user can click on the attention weights button next to each question (Fig-
Figure 3: Editing question and answers.

Figure 4: Filtering questions based on confidence score.

Figure 5: Attention weight visualization.

Figure 6: Clustering question based on different facets of the answer.

ure 3) to generate the attention weights heat map between words in the paragraph and words in the generated question (Fig. 5). Decoder attention weights represent the weights ParaQG gives to the words in the paragraph while generating the question words. For example, while decoding the question word “in”, the system gives the highest weight to the paragraph word “1909”. Similarly, the question word “disease” is generated by attending over the word “disease” in the paragraph.

Filtering and grouping questions: User can filter generated questions based on confidence scores using inter-question filter (Label 2 in figure 4) and intra-question filter (Label 1 in Fig. 4). The intra-question filter provides the user with a knob to filter questions based on the confidence score. The inter-question filter provides the user with a knob to filter low quality question-answer pairs generated from the paragraph in its entirety. We filter out unanswerable questions using our BERT-based model (explained in Section 3.4). We also group answers (and thus their associated questions) based on the stemmed form of the answer. Label 1 in Fig. 6 depicts one such answer group, whereas the generated question set is depicted by Label 2 in Fig. 6.

Editing questions with history of edits and Download generated questions and answers: If users are not satisfied with the generated question answer pairs he/she may edit it. The system stores all version of questions and answers. Users can download the final generated set of questions and answers in JSON or text format at the end.

5 Implementation Details

ParaQG¹ comprises the frontend user interface, the backend question generator and a BERT-based question filtering module. The question generator model is implemented using the PyTorch² framework. We trained the question generator model on the SQuAD 1.0 (Rajpurkar et al., 2016) dataset. We use pre-trained GloVe word vectors of 300 dimension and fix them during training. We employ a 2-layer Bi-LSTM encoder and a single-layer Bi-LSTM decoder of hidden size 600. For optimization we use SGD with annealing. We set the initial learning rate to 0.1. We train the model for 20 epochs with batch size 64. Dropout of 0.3 was applied between vertical Bi-LSTM stacks. Our question generator module provide a REST API to which we can send requests and receive responses from in JSON format. The embedded Javascript is used as the template rendering engine to render the front-end of the web application along with Bootstrap 4 for responsiveness. Express is the Web application framework used for server-side on top of

¹The source code is available for download at https://github.com/sivaanandhmuneswaran/qg-ui
²https://pytorch.org/
Node.js. For the BERT-based filtering module, we fine-tune the $BERT_{base}$ model on SQuAD 2.0 for 3 epochs, and set learning rate to 3e-5 and batch size to 12.

6 Conclusion

Question generation from text is a task useful in many application domains, yet manual generation is labour-intensive and expensive. We presented a novel online system, ParaQG, to automatically generate questions from paragraph based on pivotal answers. The system allows users to select a set of pivotal answers and then generates a ranked set of questions for each answer. ParaQG also filters out unanswerable question using a BERT-based model. ParaQG is available as a Web application, which also incorporates a novel heat map-based visualization that shows attention weights of the decoder.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th ACL*, pages 1342–1352. ACL.

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

Lynette Hirschman, Marc Light, Eric Breck, and John D Burger. 1999. Deep read: A reading comprehension system. In *ACL*, pages 325–332. ACL.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019a. Difficulty-controllable multi-hop question generation from knowledge graphs. In *ISWC*.

Vishwajeet Kumar, N. Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019b. Cross-lingual training for automatic question generation. In *ACL*.

Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. Automating reading comprehension by generating question and answer pairs. In *PAKDD*. Springer.

Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2019c. Putting the horse before the cart: A generator-evaluator framework for question generation from text. *SIGNLL Conference on Computational Natural Language Learning*. CoNLL 2019.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025.

André Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*, pages 1614–1623.

Hwee Tou Ng, Leong Hwee Teo, and Jennifer Lai Pheng Kwan. 2000. A machine learning approach to answering questions for reading comprehension tests. In *SIGDAT-EMNLP. ACL-Volume 13*, pages 124–132. ACL.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you dont know: Unanswerable questions for squad. In *ACL*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1605.02650*.

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. *arXiv preprint arXiv:1603.06807*.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *NAACL-HLT*, volume 2, pages 569–574.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

John H Wolfe. 1976. Automatic question generation from text-an aid to independent study. *ACM SIGCSE Bulletin*, 8(1):104–112.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *EMNLP 2018*, pages 3901–3910.