Question Generation Using Sequence-to-Sequence Model with Semantic Role Labels

Alireza Naeiji¹, Aijun An¹, Heidar Davoudi², Marjan Delpisheh¹, Muath Alzghool¹
¹Department of Electrical Engineering and Computer Science, York University, Canada
²Faculty of Science, Ontario Tech University, Canada
{naeiji, aan, mdelpishe, alzghool}@yorku.ca
heidar.davoudi@ontariotechu.ca

Abstract

Automatic generation of questions from text has gained increasing attention due to its useful applications. We propose a novel question generation method that combines the benefits of rule-based and neural sequence-to-sequence (Seq2Seq) models. The proposed method can automatically generate multiple questions from an input sentence covering different views of the sentence as in rule-based methods, while more complicated “rules” can be learned via the Seq2Seq model. The method utilizes semantic role labeling to convert training examples into their semantic representations, and then trains a Seq2Seq model over the semantic representations. Our extensive experiments on three real-world data sets show that the proposed method significantly improves the state-of-the-art neural question generation approaches.

1 Introduction

Question Generation (QG) from text has gained increasing interest due to its usefulness in various applications such as educational reading comprehension assessment (Chen et al., 2018; Kumar et al., 2018), data augmentation for training question-answering systems (Sultan et al., 2020), and response generation in conversational systems (Gu et al., 2021). We have been working on QG for the purpose of automatically creating the knowledge base (KB) for a conversational QA system of an industry partner¹. The knowledge base consists of QA pairs extracted from a domain-specific document (such as car manuals). Previously, the creation of their KB was done manually, which is very labor-intensive. To automate this KB generation process, we have tried both rule-based and neural sequence-to-sequence (Seq2Seq) QG methods.

The rule-based methods create rules based on linguistic features that capture the relationships among components of a sentence and can generate multiple questions from an input sentence to cover different aspects of the sentence. However, designing such rules is a very labour-intensive task as well. Also, these rules may not capture the complexity of ways a human asks questions (Yuan et al., 2019). As we will show in Section 4, the rule-based method does not lead to good results compared to neural Seq2Seq models.

While the Seq2Seq methods achieve better results, such methods are highly data-driven. For domains with limited training data (such as car manuals), relations that map the input text to questions cannot be well captured. In addition, Seq2Seq models often generate a single question from an input text. However, multiple questions can be asked about a piece of text from different aspects. One way to generate multiple questions with Seq2Seq models from an input text is to mark the input text with different answer spans or keywords to show the focus for QG so that multiple questions may be generated from the same text, one for each answer span/keyword. However, in our application, such answer spans or keywords are not available as marking answer spans or keywords when creating training data requires intensive labor work. Our partner prefers an answer-unaware QG system that can automatically generate multiple factual questions without indicating answer spans or keywords in either training or inference time. Another technique for generating multiple questions is to use diverse beam search (Vijayakumar et al., 2018) However, the beam search methods require the user to specify the number of questions returned, which is hard to specify as the ideal number of generated questions varies among different input texts.

To address these issues, we propose a novel approach to question generation, which uses semantic role labeling (commonly used in rule-based systems) that can label an input sentence in different ways corresponding to multiple semantic views of...
an input sentence, and then trains a Seq2Seq model with SRL-labeled sequences for question generation. Our method does not need keyword/answer span labels in the training data nor specifications of the number of multiple answers to be generated. The use of SRL also increases the number of training examples, which may help alleviate the problem of the limited labeled data problem.

We evaluate the proposed method on three real-world data sets and compared the proposed method with several state-of-the-arts QG methods including the ones that generate multiple questions. The extensive experiments on these data sets show that the proposed framework is significantly better than the state-of-the-art Seq2Seq models and rule-based methods, especially in terms of coverage and overall scores considering both precision and recall.

2 Related Work

The question generation approaches broadly fall into two categories: rule-based approaches and neural Seq2Seq learning approaches. Rule-based approaches mainly rely on hand-crafted templates/rules built upon linguistic features (Chali and Hasan, 2015; Flor and Riordan, 2018; Khullar et al., 2018; Lindberg et al., 2013). These methods use rigid heuristic rules to transform a source sentence into one or more questions. However, rules have limited power in expressing the complicated mapping function that the human uses for question generation. Designing a comprehensive set of rules is a very labour-intensive task.

Recently, neural Seq2Seq models have been successfully applied to question generation due to its capability to extract effective features and model complicated functions. Early Seq2Seq-based QG models are based on RNN structures. Examples include an LSTM-based Seq2Seq model with the global attention mechanism (Du et al., 2017) and LSTM-based model with the maxout pointer and gated self-attention network (MP-GSN) (Zhao et al., 2018). More recent Seq2Seq models are based on Transformer which relies entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs. State-of-the-art Transformer-based models that have been pre-trained for QG include T5 (Raffel et al., 2020), BART (Lewis et al., 2020), ProphetNet (Qi et al., 2020), UNILM (Dong et al., 2019), UNILMv2 (Bao et al., 2020), and Ernie-Gen (Xiao et al., 2020), to list some examples.

While the Seq2Seq methods achieve better results than rule-based methods (Du et al., 2017), they are highly data-driven. In addition, Seq2Seq models often generate a single question given an input text, which does not cover multiple views of a sentence.

To solve this single-question-generation problem, different strategies have been proposed. One strategy is to use diverse beam search (Vijayakumar et al., 2018; Zhang and Zhu, 2021) or sampling techniques (such as top-\(p\) nucleus sampling used in (Sultan et al., 2020)). While these methods showed promising results, the user has to specify the bin/sample size and the number of questions to be generated (whose ideal number may depend on the input sequence). Another strategy for generating diverse questions is to mark or extract keywords in the input text and generate questions by conditioning on keywords or keyword positions (e.g., (Pan et al., 2020; Shen et al., 2020; Subramanian et al., 2018; Sun et al., 2018; Song et al., 2018; Zhang and Zhu, 2021)). However, extracting keywords (either automatically or manually) to build keyword-labeled training data often needs domain knowledge, a pre-defined keyword list, or documents beyond the training data. The QG method we propose solves these problems by learning Seq2Seq models with semantic role labeled QAs. It can generate multiple and diverse questions without specifying the number of questions to be generated or requiring keyword-labeled training data.

A recent method that also uses SRL and Seq2Seq models is a 2-step method in (Pyatkin et al., 2021). It tackles role question generation that, given a predicate mention and a passage, generates a set of questions asking about all possible semantic roles of the predicate. It first generates prototype questions for all the roles based on the ontology in PropBank (Palmer et al., 2005). It then trains a BART model to generate all questions (including ones that cannot be answered by the input text) given these prototype questions contextualized over the input text. Both the problem definition and the methodology are very different from ours. Our method does not need to generate prototype questions and we generate only the information-seeking questions that can be answered by the input text.

3 Proposed Methodology

Given a set of answer (i.e., sentence) and question pairs, our goal is to train a model to generate from an unseen sentence one or more questions that can
be answered by the sentence.  

3.1 Overview of the Method

Our method contains a Semantic Role Labeler (SRLer), a Seq2Seq model, and two semantic mappers (namely, Question2SRL and SRL2Question). First, SRLer extracts semantic representations (i.e., SRL labels) from answers in the training set. Then Question2SRL maps questions in the training set to their corresponding semantic representations. Next, a Seq2Seq model is trained using these semantic representations to convert an SRL representation of an answer to that of a question. In the inference stage, Semantic Role Labeler extracts semantic representations ($\alpha_{\text{sem}}$) of an answer $\alpha$. Then, $\alpha_{\text{sem}}$ is converted to an SRL representation of a question ($\hat{q}_{\text{sem}}$) by the learned Seq2Seq model. Finally, SRL2Question converts $\hat{q}_{\text{sem}}$ into a natural language question $\hat{q}$.

Figure 1: Overview of Proposed Framework

3.2 Semantic Role Labeler

A Semantic Role Labeler (SRLer) is used in both training and inference. In the training phase, an SRLer is used to convert each answer $\alpha$ in the training set into its semantic representation (denoted as $\alpha_{\text{sem}}$) that contains semantic role labels (SRL).

An SRLer recognizes the predicate-argument structure of a sentence and assigns labels (i.e., semantic roles, such as agent, goal or result) to words or phrases in a sentence. We use SRL BERT (Shi and Lin, 2019) provided in AllenNLP (Gardner et al., 2017) to produce the semantic labels for the input text, which is the state-of-the-art model for SRL extraction. This method generates a predicate-argument structure for a sentence based on a BERT-based approach. In this model, each sentence is represented by one or more propositions, consisting of a predicate (usually a verb) and its semantic arguments. For example, the semantic representation of sentence “ABS is activated during braking under certain road or stopping conditions” is “[ARG1] is activated [ARGM-TMP]”, where [ARG1] (representing patient) and [ARGM-TMP] (representing time) are semantic role labels for ABS and during braking under certain road or stopping conditions, respectively.

Table 1 provides more examples of semantic representations for answers. Note that if a sentence contains more than one verb, more than one semantic representation may be generated. For example, S3 and S4 in Table 1 are from the same sentence.

3.3 The Question2SRL Mapper

The Question2SRL mapper converts a question $q$ in the training data into its semantic representation $q_{\text{sem}}$. Instead of applying an SRLer directly on $q$, Question2SRL uses the semantic role labels in the semantic representation $\alpha_{\text{sem}}$ of $q$’s corresponding answer $\alpha$ to label the phrases or words in $q$. We design two approaches for Question2SRL, namely, Hard-Question2SRL and Soft-Question2SRL:

**Hard-Question2SRL**: In this approach, for each semantic role label $l$ that occurs in an answer’s SRL representation, if its corresponding phrase or word occurs in the question, the phrase or word in the question is replaced with label $l$. The reason why we did not use semantic role labeling to directly label the question is that we would like to keep the question words (e.g., what, where, when, etc.) in the semantic representation of the question, and also that semantic role labeling may generate labels for a question which do not occur in its answer. The lower part of Table 1 provides the semantic representations of the questions corresponding to the answer SRL representations in the upper part of the table. Note that Q3 and Q4 are two different representations of the same question, resulting from two different SRL representations of the same answer (S3 and S4). Thus, one original training example

---

2 We work on sentence-level QG instead of paragraph-level QG because the paragraphs in our application domain (car manuals) are often short and contain a single sentence. Also, an answer sentence in our car manuals dataset does not contain answer words labels, which is different from most paragraph-level QG systems where answer words/phrases are marked in an input paragraph. Sentence-level QG can be extended to paragraph-level QG via sentence segmentation after coreference resolution.

3 We also used the Clear Parser SRL (Choi and Palmer, 2011) in our experiments. SRL BERT leads to better results. In this paper we report the results from SRL BERT.
Table 1: Semantic representation of answers and questions (ARG0: agent, ARG1: patient, ARG2: attribute, ARGM-NEG: negation, ARGM-PRP: purpose)

Table 2 shows how the algorithm works using an example. The first box of the table illustrates the answer $a$, and its respective question $q$. Each subsequent box shows a semantic role label $l \in L$ in $a_{sem}$, the best n-gram $ng$ matching with $words(l)$, and their respective similarity score. The final box shows the semantic representation of question $q_{sem}$ after replacing n-grams with SRLs in previous steps. Note that we replace the n-gram with an SRL label if the score is higher than or equal to a threshold (i.e., $\alpha$).

**Algorithm 1: Soft-Question2SRL**

| Input | $L$ // a set of SRLs generated for $a$  
| Output | $q_{sem}$ // semantic rep. of $q$  
| $nG$ | Generate all n-grams from $q$  
| for each $l \in L$ do |  
| $score$ | $0$  
| for each $ng \in nG$ do |  
| $sim$ | $\text{Cosine}(words(l), ng)$  
| if $score < sim$ then |  
| $score$ | $\text{replace } ng \text{ in } q \text{ with } l_{best}$  
| if $score \geq \alpha$ then |  

3.4 Sequence-to-Sequence Learning

Given a set of $(a_{sem}, q_{sem})$ pairs (where $a_{sem}$ and $q_{sem}$ are an SRL-labeled answer and an SRL-labeled question, respectively), we train a Seq2Seq model to convert $a_{sem}$ to $q_{sem}$. Any Seq2Seq model can be used for this purpose. In our experiment, we use the state-of-the-art models T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) to evaluate our method.

We feed the Seq2Seq model with $a_{sem}$ as the input sequence, where some of words/phrases in the original answer $a$ are replaced by SRLs. While SRLs provide the Seq2Seq model with useful information, this replacement may reduce the information to which model is exposed. As another strategy, we add the actual answer $a$ as a context to the input sequence. That is, the input to the Seq2Seq model is $(answer: a_{sem} context: a)$, where $answer: and context: are tokens prepended to $a_{sem}$ and $a$, respectively. Alternatively, we can add the SRL labels of the actual answer $a$ and their corresponding
words separated by a special token <sep> to the input sequence. The input to the Seq2Seq model is \textlangle answer: a_{sem} <sep> label words <sep> \textrangle.

Table 3 shows a training example with three variations of the input. We will compare the three input versions of the Seq2Seq model in the experiment.

3.5 The SRL2Question Mapper

In the inference phase, the SRLer is first used to convert an input sentence (i.e., answer $\hat{a}$) into its semantic representation ($\hat{a}_{sem}$). Then, the trained Seq2Seq model is used to convert $\hat{a}_{sem}$ into a semantic representation of a question ($\hat{q}_{sem}$). After that, the SRL2Question mapper transforms all the semantic role labels in the generated semantic representation $\hat{q}_{sem}$ into words or phrases. In particular, for each semantic role label $l$ in the semantic representation $\hat{q}_{sem}$ generated by the Seq2Seq model, the SRL2Question mapper looks for label $l$ in all the semantic representations $\hat{a}_{sem}$ of the input sequence $\hat{a}$, and uses the phrase or word in $\hat{a}$ that corresponds to $l$ to replace $l$ in $\hat{q}_{sem}$. Table 4 shows examples of generated semantic representations and converted questions, together with their input sentences, semantic representations of the input sentences, and ground truth questions.

4 Empirical Evaluation

We investigate (1) whether using SRL with Seq2Seq models improves the QG performance of Seq2Seq without SRL, and (2) how our method compares with the state-of-the-art QG methods.

4.1 Datasets

We evaluate our method on 3 datasets\(^4\). The first one contains QAs created by human annotators from two car manuals of Ford and GM, denoted as Car Manuals. The second dataset is SQuAD (Rajpurkar et al., 2016), containing QAs created by Amazon Mechanical Turk crowd-workers from Wikipedia articles. We use the processed sentence-level SQuAD dataset (Du et al., 2017), where the answer in a QA pair is a single sentence. The third dataset is NewsQA (Trischler et al., 2016), a machine comprehension dataset of human-generated QA pairs from CNN news articles. We created a sentence-level NewsQA dataset. The sentences were extracted from corresponding paragraphs based on their answer span. All the datasets contain training, testing and development sets. Their statistics are given in Table 5. In these datasets, multiple QA pairs may have the same answer sentence but different questions. For example, two QA pairs in SQuAD share “alfred north whitehead was born in ramsgate, kent, england, in 1861” as the answer, but their questions (i.e., “in what year was whitehead born?” and “where was alfred north whitehead born?”) are different and cover different aspects of the answer sentence.

4.2 Automatic Evaluation Metrics

For automatic evaluation, we use the precision, recall and F scores proposed in (Schlichtkrull and Cheng, 2020)\(^5\) for measuring the quality and diversity of generated questions. Given a test example \(q_{sem} = \text{ARGM-TMP} \), who named \text{ARG1} \text{ARG2} ?

\(^4\)The code and data sets are available at https://github.com/Naeiji/QGwSRL

\(^5\)These scores are called $u$, $v$ and $F$ scores in (Schlichtkrull and Cheng, 2020).

Table 2: An example of Soft-Question2SRL mapper converting a question into its semantic representation. Each row shows the best score and corresponding n-gram in the question ($\alpha = 0.85$).
Table 3: A training example for Seq2Seq model with 3 different strategies for input (i.e., answer) representations.

<table>
<thead>
<tr>
<th>input</th>
<th>answer: [ARG1] was named [ARG2], [ARGM-PRD].</th>
<th>input+L</th>
<th>answer: [ARG1] was named [ARG2], [ARGM-PRD]. &lt;sep&gt; [ARG1] denver linebacker von miller &lt;sep&gt; [ARG2] super bowl mvp &lt;sep&gt; [ARGM-PRD] recording five solo tackles , 2 1/2 sacks , and two forced fumbles.</th>
<th>input+L</th>
<th>answer: [ARG1] was named [ARG2], [ARGM-PRD].</th>
<th>context: denver linebacker von miller was named super bowl mvp, recording five solo tackles, 2 1/2 sacks, and two forced fumbles.</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>who won the [ARG2] ?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Examples of sentence $\hat{q}$, its semantic representation $\hat{q}_{\text{sem}}$, the outcome $\hat{q}_{\text{sem}}$ generated by Seq2Seq, the question $q$ converted from $\hat{q}_{\text{sem}}$, and ground-truth question $Q_t$ from the Car Manuals dataset.

| $\alpha$:  | before placing a child in the child restraint , make sure it is securely held in place . |
| $\hat{q}_{\text{sem}}$:  | before placing [ARG1] [ARG2] , make sure it is securely held in place . |
| $\hat{q}$:  | what should i do before placing [ARG1] [ARG2] ? |
| $Q_t$:  | what should i do before placing a child in the child restraint ? |

| $\hat{\alpha}_{\text{sem1}}$:  | adjust the temperature setting using the + and - temperature buttons on the right-hand side of the climate controls . |
| $\hat{\alpha}_{\text{sem2}}$:  | adjust the [ARG1] setting using the + and - temperature buttons on the right-hand side of the climate controls . |
| $\hat{q}_{\text{sem1}}$:  | how do i adjust the [ARG1] setting ? |
| $\hat{q}_{\text{sem2}}$:  | how do i adjust the temperature [ARGM-LOC] ? |
| $\hat{q}$:  | how do i adjust the temperature setting ? |
| $Q_t$:  | how do i adjust the temperature on the passenger ’s side ? |

We conducted experiments with two state-of-the-art transformer-based Seq2Seq models: (1) BART (Lewis et al., 2020) and (2) T5 (Raffel et al., 2020).

We fine-tune the Huggingface pretrained models (Wolf et al., 2019) of BART-base and T5-small with 139 and 60 million parameters respectively. We use the smallest available model sizes of BART and T5 to avoid GPU memory error. Four V100-SXM2 32GB GPUs are used for fine-tuning. For baselines, T5 and BART are fine-tuned using the original sentence-based QA pairs in each training set to generate a question given an answer.

For our method, we use a pre-trained SRL BERT model (Shi and Lin, 2019) provided in AllenNLP (Gardner et al., 2017) as the Semantic Role Labeler to convert answer sentences to their SRL representations. We then fine-tune T5 or BART to learn a model that maps an SRL representation of an input sentence to that of the question, which is then mapped to a natural language question.

To determine the values of hyperparameters in the Seq2Seq model (T5 and BARR) in either baseline or our model, we conduct random search (Bergstra and Bengio, 2012) due to its efficiency. To do so, we randomly select 10 combinations of learning rates (LR) and epoch numbers (EP) as follows: "LR = 5 \times 10^{-6}, EP = 2"; "LR = 10^{-5}, EP = 8"; "LR = 5 \times 10^{-5}, EP = 5"; "LR = 5 \times 10^{-5}, EP = 10"; "LR = 10^{-4}, EP = 10"; "LR = 10^{-4}, EP = 4"; "LR = 5 \times 10^{-4}, EP = 3"; "LR = 5 \times 10^{-4}, EP = 5".
EP = 7”; "LR = 10−4, EP = 4”; and "LR = 10−4 EP = 10”. The best model for each dataset and each method is chosen based on the final loss of the development set. We combine an actual batch size of 16 with 2 gradient accumulation steps per minibatch to artificially create a batch size of 32. The maximum sequence length is set to 512 for both input and output. For decoding method, we use beam search with a beam size of 10.

4.3.1 Results of Automatic Evaluation

Table 6 shows the automatic evaluation results on Car Manuals with BART and T5 as the baseline. **Hard** is our method using **Hard-Question2SRL** for **Question2SRL**. **Soft** is our method with **Soft-Question2SRL** with 80% as the soft-matching threshold (α). **Soft+C** and **Soft+L** are our methods with **Soft-Question2SRL** plus using the original question or SRL labels and their corresponding words as the context, respectively.

Table 6: Automatic evaluation results on Car Manuals with BART and T5 as Baselines (P, R and F mean Precision, Recall and F-score in %). Hard and Soft+ methods are variations of our method with BART or T5 as the Seq2Seq model.

As shown in Table 6, using SRL representations to train a QG model with either BART or T5 significantly improves the baseline performance on Car Manuals. This is due to generalization of sentences into SRL representations, allowing general semantic patterns to be modeled and used in QG. The use of SRLs also increases the number of training examples due to the fact that the SRLer can convert a sentence into multiple SRL-labeled sentences. All the variations of our method significantly increase the recall and F-scores in all types of measures (BLEU, ROUGE and METEOR) and they also increase precision in almost all of the cases. The recall increase is due to the fact that the SRLer can convert a sentence into multiple SRL-labeled sentences, leading to questions asking about different aspects of the sentence.

Comparing the hard and soft versions of our method, **Soft-Question2SRL** is better than **Hard-Question2SRL** in all cases, indicating that allowing different but similar expressions in the corresponding question and answer when labelling the question with SRLs is important and beneficial. We also see that the use of the original source sentence or SRL labels as a context in the input is beneficial. This is probably because the error propagation from semantic role labeling has less impact when (part of) the original sentence are given as a context.

Table 7: Automatic evaluation results on SQuAD with BART and T5 as baselines (P, R and F mean Precision, Recall and F-score). Hard and Soft+ methods are different variations of our method with BART and T5 as the Seq2Seq models.

Tables 7 shows the automatic evaluation results on the SQuAD dataset with BART and T5 as the baseline. Again, we observe that all variations of our method significantly outperform the baseline on recall and F-score. On this data set, the BART baseline shows the best precision. However, the lower precision of our method is due to the incompleteness of the ground truth questions in SQuAD. That is, many of the questions our method generates are good questions, but they do not match the ground truth questions in the data set.

Table 11 in Appendix B shows the generated questions for 3 testing examples from our method (Soft+L) with T5 and T5 without SRL. As shown, our method generates more questions covering different aspects of an input sentence, while T5 without SRL generates only one question. In all the 3 examples, there is only one ground-truth question. When precision is computed, the extra questions we generated have a low precision due to poor match with the ground truth even though they are good questions. This explains why our methods have lower precision scores than the baselines.

The results on this dataset also show that soft matching is better than hard matching, and the use of context is better with T5, but has no obvious

---

6Such a problem is also mentioned by others such as in (Sultan et al., 2020)
advantage for BART. We show the impacts of soft-matching threshold $\alpha$ in Appendix E. In practice, $\alpha$ can be tuned using the development set.

Similar results are observed on the NewsQA dataset, as shown in Table 12 in Appendix C.

### 4.3.2 Human Evaluation

We randomly selected 50 input sentences from the SQuAD dataset and asked 5 English speakers to rate the quality of generated questions from each method in terms of recall, clarity, Q&A relatedness and grammar. All the criteria are rated based on a 5-point scale. A precision score for each question is computed by averaging the scores for clarity, Q&A relatedness and grammar. An overall score (F score) for each test input sentence is computed as the harmonic mean of precision and recall scores. Detailed information on the questionnaire we provided to the human evaluators is provided in Appendix 4.3.2.

Table 8 shows the average scores among the human evaluators. The results show that our methods (Soft+L with BART and T5+Soft with T5) are significantly better than their corresponding baseline in recall and F-measure, and they are also better than the baselines in precision.

### 4.4 Comparison with other SOTA methods

We compare our method with additional SOTA baselines, most of which can generate multiple questions from an input sentence:

- **BART-multi** and **T5-multi**. BART or T5 model fine-tuned to generate multiple sequences given an input sequence by using `<sep>` tokens to separate the ground-truth questions in the output part of each training example.

- **BART-divbeam** and **T5-divbeam**. BART or T5 model that uses decoding-based diverse beam search (Vijayakumar et al., 2016) to generate multiple questions. We use beam size of 6 with 3 diverse groups and diversity penalty of 0.4 all same as in (Zhang and Zhu, 2021). We select 3 best questions for the evaluation.$^7$

- **ProphetNet-single** and **ProphetNet-multi**. ProphetNet (Qi et al., 2020) is another Transformer-based SOTA model. In ProphetNet-single, ProphetNet is fine-tuned to generate a single question. In ProphetNet-multi, it is fine-tuned to generate multiple sequences from an input sequence by using `<sep>` tokens to separate the ground-truth questions.

- **MP-GSN** (Zhao et al., 2018) An LSTM-based Seq2Seq QG model. We use the sentence-level MP-GSN with the default setting in the implementation of MP-GSN in (Lee, 2019).

- **Rule-based method** with 75 rules based on semantic role labels to convert a sentence into questions.$^8$

The hyper-parameters for all the Seq2Seq methods are determined through random search (described in Section 4.3).

Table 9 shows the results of these SOTA methods on SQuAD compared to our method (Soft+L) with BART or T5 as the Seq2Seq model. As shown, our methods outperform all the baselines in F-score and recall. It is also the best in precision in all underlying metrics except for precision using METEOR where ProphetNet-single is the best and our methods are the second best.

Compared to Seq2Seq-multi, our methods outperform them significantly in all metrics, indicating using SRL to generate multiple questions is much more effective than using the `<sep>` tokens in the output parts of the training sequences.

Our methods also outperform the diverse-beam-search-based methods for generating multiple questions, indicating different SRL representations of a sentence can more effectively lead to generation of diverse questions that cover different aspects of the input sentence.

Compared to the rule-based method, our method outperforms it significantly on all measures with big margins. This indicates that SRL-based Seq2Seq model better captures the relations between SRL representations of questions and those of answers than rules, which complicated “rules” can be learned by Seq2Seq models.

### 5 Performance with Different Data Sizes

Among the 3 datasets, Car Manuals is the smallest, and the improvements of our method on this dataset over the baselines are the largest. To further investigate whether our method makes better improvement when the data set is small, we conducted an experiment with one quarter of the SQuAD dataset.$^8$

---

$^7$An example rule is “Replace [ARG1] with what if it appears at the beginning of the sentence”.

$^8$An example rule is “Replace [ARG1] with what if it appears at the beginning of the sentence”.

---

2837
Table 8: Human evaluation results on 50 test sentences from SQuAD. Precision is the average of Clarity, Relatedness and Grammar scores. F-measure is the harmonic mean of precision and recall scores.

<table>
<thead>
<tr>
<th>QG System</th>
<th>F-measure Score (1-5) ± stdev</th>
<th>Precision Score (1-5)</th>
<th>Recall Score (1-5)</th>
<th>Clarity Score (1-5)</th>
<th>Relatedness Score (1-5)</th>
<th>Grammar Score (1-5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BART</td>
<td>3.64 ± 0.33</td>
<td>4.73</td>
<td>3.06</td>
<td>4.70</td>
<td>4.56</td>
<td>4.94</td>
</tr>
<tr>
<td>Soft+L (with BART)</td>
<td>4.32 ± 0.28</td>
<td>4.77</td>
<td>4.16</td>
<td>4.75</td>
<td>4.61</td>
<td>4.93</td>
</tr>
<tr>
<td>T5</td>
<td>3.63 ± 0.37</td>
<td>4.71</td>
<td>3.08</td>
<td>4.66</td>
<td>4.63</td>
<td>4.65</td>
</tr>
<tr>
<td>Soft+L (with T5)</td>
<td>4.40 ± 0.29</td>
<td>4.73</td>
<td>4.25</td>
<td>4.72</td>
<td>4.48</td>
<td>4.90</td>
</tr>
</tbody>
</table>

Table 9: Comparison with SOTA models on SQuAD

<table>
<thead>
<tr>
<th>QG Method</th>
<th>BLEU-4 ±</th>
<th>ROUGE-L</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5-multi</td>
<td>20.0 ± 0.0</td>
<td>22.0 ± 0.0</td>
<td>26.0 ± 0.0</td>
</tr>
<tr>
<td>BART-pre-trained</td>
<td>21.0 ± 0.0</td>
<td>22.0 ± 0.0</td>
<td>28.0 ± 0.0</td>
</tr>
<tr>
<td>ProphetNet-single</td>
<td>21.0 ± 0.0</td>
<td>22.0 ± 0.0</td>
<td>28.0 ± 0.0</td>
</tr>
<tr>
<td>ProphetNet-biased</td>
<td>21.0 ± 0.0</td>
<td>22.0 ± 0.0</td>
<td>28.0 ± 0.0</td>
</tr>
<tr>
<td>Rule Based</td>
<td>21.0 ± 0.0</td>
<td>22.0 ± 0.0</td>
<td>28.0 ± 0.0</td>
</tr>
<tr>
<td>Soft+L (with BART)</td>
<td>23.0 ± 0.0</td>
<td>25.0 ± 0.0</td>
<td>30.0 ± 0.0</td>
</tr>
<tr>
<td>Soft+L (with T5)</td>
<td>23.0 ± 0.0</td>
<td>25.0 ± 0.0</td>
<td>30.0 ± 0.0</td>
</tr>
</tbody>
</table>

Figure 3 in Appendix F illustrates the relative improvement of our method over the T5 baseline in F-measure, precision and recall. The relative improvement is computed as: \( \frac{\text{Improvement}}{\text{Score}_{\text{T5}}} \). The blue bars in the figure represent the improvement on the one-quarter subset of SQuAD and yellow ones represent that on the whole SQuAD dataset. Clearly, the improvements of our method over the T5 baseline are larger on the smaller dataset than on the whole SQuAD dataset. This indicates that our SRL-based Seq2Seq method can better handle smaller datasets than its Seq2Seq baselines due to the increase in training examples when we label the QAs with SRLs.

6 Conclusions

We proposed a novel QG method that learns a Seq2Seq model to convert an SRL representation of an input sentence into an SRL representation of a question, which is then converted to a natural language question. Similar to rule-based methods, our SRL-based Seq2Seq method can generate multiple questions from an input sentence, significantly improving the recall and overall performance of Seq2Seq QG. It is also much better than rule-based methods because better and more complicated “rules” can be learned via the Seq2Seq model. Our evaluation on three real-world datasets shows that the proposed method significantly outperforms both rule-based, original Seq2Seq methods and several other SOTA models, especially in recall and overall performance. As future work, we will extend this method to paragraph-based question generation.

Limitations

A limitation of our method is that its performance depends on the SRL performance. If the input sentences are not well-formed, semantic role labeling may not produce correct labels. Also, we found that some of the questions generated by our method for the same input sentence may be similar or same in meaning with some minor differences in the use of words. This may not be a problem if the application allows similar questions to be generated (e.g., for reading compression). But in the application where duplicated questions are not allowed or not desired, a post-processing step to remove questions with the same meaning is needed.

Acknowledgements

We would like to thank iNAGO Corporation for providing the Car Manuals dataset used in this research and for their collaboration on the research topic. This work is supported by an NSERC Alliance grant from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References


Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. 2018. Learningq: a large-scale dataset
for educational question generation. In Twelfth International AAAI Conference on Web and Social Media.


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


Sheng Shen, Yaliang Li, Nan Du, X. Wu, Yusheng Xie, Shen Ge, Tao Yang, Kai Wang, Xingzheng Liang, and Wei Fan. 2020. On the generation of medical question-answer pairs. In AAAI.


Table 10 provides the average number of questions generated by different versions of our methods on the SQuAD dataset.

Table 11 shows 3 examples of an input sentence $\tilde{a}$, its ground-truth question ($q_g$) and the generated questions ($\hat{q}_i$) from our method (Soft+C) with T5 compared to the question ($g_t$) generated from the T5 baseline without the use of SRLs. As shown, our method generates more questions covering different aspects of the input sentence, while T5 without SRL generates only one question. In all these 3 examples, there is only one ground-truth question in the data set. When precision is computed, the extra precision for our method is due to the additional questions generated by our method.
Table 11: Three Examples showing the input sentence \( \hat{a} \), its ground-truth question \( q_g \), the question generated by T5 (\( q_t \)), and the questions \( \hat{q}_i \) generated by our Soft+L method with T5 and alpha=80% on the SQuAD dataset.

\[
\begin{array}{|c|c|c|c|c|}
\hline
\hat{a}: & \text{there are two categories of repetitive dna in genome: tandem repeats and interspersed repeats.} \\
q_g: & \text{what are two types of repetitive dna found in genomes?} \\
\hat{q}_1: & \text{what are the two categories of repetitive dna in genome?} \\
\hat{q}_2: & \text{how many categories of repetitive dna are there in the genome?} \\
\hat{q}_3: & \text{before the solar/wind revolution, Portugal had generated electricity from hydropower plants on its rivers for decades.} \\
q_g: & \text{through what renewable resource had Portugal generated electricity before the solar/wind revolution?} \\
\hat{q}_1: & \text{what had Portugal generated electricity from before the solar/wind revolution?} \\
\hat{q}_2: & \text{how long had Portugal generated electricity from hydropower plants on its rivers?} \\
\hat{a}: & \text{the term parinirvana is also encountered in Buddhism, and this generally refers to the complete nirvana attained by the arahant at the moment of death, when the physical body expires.} \\
q_g: & \text{what term is used for the complete nirvana attained by the arahant at death?} \\
\hat{q}_1: & \text{what is the term parinirvana used in Buddhism?} \\
\hat{q}_2: & \text{what term is also encountered in Buddhism?} \\
\hat{q}_3: & \text{when is the complete nirvana attained by the arahant?} \\
\hat{q}_4: & \text{who attained the complete nirvana at the moment of death?} \\
\hat{q}_5: & \text{what does the term parinirvana refer to at the moment of death?} \\
\hat{q}_6: & \text{what term is used in Buddhism to describe the complete nirvana attained by the arahant at the moment of death?} \\
\end{array}
\]

For Recall, the ratings are:
- 1= Bad: the generated questions do not cover any fact in the answer;
- 2= Unacceptable: the generated questions cover only a small portion of the facts in the answer;
- 3= Borderline: the generated questions cover around 50% of the facts in the answer;
- 4= Acceptable: the generated questions cover most of the facts in the answer; and
- 5= Good: the generated questions cover all the facts in the answer.

For Clarity, the ratings are:
- 1= Bad: the question is completely unclear in meaning or makes no sense;
- 2= Unacceptable: the question is mostly unclear;
- 3= Borderline: the question is between unacceptable and acceptable;
- 4= Acceptable: the question is clear and understandable, but the use of words can be improved; and
- 5= Good: the question has no problem. It is clear, simple and uses the right words.

For Q&A Relatedness, the ratings are:
- 1= Bad: the question is completely unrelated to the answer sentence it is generated from;
- 2= Unacceptable: the question is somewhat related to the answer sentence, but it cannot be answered by the answer sentence;
- 3= Borderline: the question can be partially answered by the answer sentence, but far from completely;
- 4= Acceptable: the question can be mostly answered by the answer sentence, although maybe not completely; and
- 5= Good: the question can be very well answered by the answer sentence.

For Grammar, the ratings are:
- 1= Bad: the grammar of the question is completely wrong;
Figure 2: BLEU-4, ROUGE-L and METEOR F-scores of T5 on SQuAD (SQ) and Car Manuals (AM) using different alphas values.

• 2= Unacceptable: the question has major grammatical problems;
• 3= Borderline: the question has a grammatical error, which is between major and minor;
• 4= Acceptable: the question has only a minor grammatical problem; and
• 5= Good: the question is completely grammatically correct.

E Sensitivity Analysis

To see how the soft-matching threshold $\alpha$ affects the results, we experiment with different $\alpha$ values ranging from 70 to 95 and use the resulting datasets to fine-tune T5. In this experiment, we use the best configurations of the T5 model on the Car Manuals and SQuAD datasets. Six $\alpha$ values of 70, 75, 80, 85, 90, and 95 are selected for this purpose. Figure 2 shows how the F-score of BLEU-4, ROUGE-L and METEOR changes with $\alpha$ on SQuAD (SQ) and Car Manuals (CM). As can been seen, on SQuAD all the lines are quite flat, indicating $\alpha$ values do not have much impact on the performance. This is likely due to the best-matching n-gram in a ground-truth question either matches with the word/phrase replaced by an SRL in the answer very well or very poorly, leading to insensitivity to the threshold value between 70 and 95. Similar results are observed on the Car Manuals dataset although the scores are increasing with the $\alpha$ on this dataset, but slowly.

F Performance vs Different Data Sizes

To further investigate whether our method makes better improvement over the baselines when the data set is small, we conducted an experiment with one quarter of the SQuAD dataset. Figure 3 illustrates the relative improvement of our method over the T5 baseline in F-measure, Precision and Recall. The relative improvement is computed as: $rac{\text{score}_{\text{our}} - \text{score}_{\text{T5}}}{\text{score}_{\text{T5}}}.$

The blue bars in the figure represent the improvement on the one-quarter subset of SQuAD and yellow ones represent that on the whole SQuAD dataset. Clearly, the improvements of our method over the T5 baseline are larger on the smaller dataset than on the whole SQuAD dataset. This indicates that our SRL-based Seq2Seq method can better handle smaller datasets than its Seq2Seq baselines due to the increase in training examples when we label the QAs with SRLs.