A Reference-free Method Evaluating the Quality of Generated Questions

To reproduce the results or generated new sorted questions, you can refer to "`tryout_project/README.md`".

Abstract

As QG methods show a promising ability on generating questions from texts, how to evaluate the generated questions has attached much attention. However, the lack of annotated dataset and effective metrics makes it hard to make a fair assessment. In this project, I propose a method that evaluates the questions from 3 aspects: 1) grammar and fluency; 2) answerability; 3) deductibility. Questions are sorted according to the weighted sum of scores of each dimension, which indicates the quality of questions. The results of experiments and case studies show that my method does a good job of evaluating questions.

Problem Analysis

The goal of the project is that given several question-answer pairs(QA-pairs) generated from given stories, I need to evaluate the questions without any annotation, and sort them based on their quality.

Before doing the survey, it's worth figuring out what is a good question. I define a good question from 3 aspects:

First, a good question should not have grammatical errors and should be semantically fluent.

Second, a good question can be correctly answered according to the given context. This is important when questions are generated from stories, because the background of a story may be fictional. In this case, questions can only be correctly answered based on context instead of factual knowledge or commonsense. This is referred to as "answerability".

Third, the relationship between questions and answers ought to be inferrfrom the context. Both questions and answers shouldn't be inconsistent with the document, or ambiguous to respondents.

Of course, there are other opinions about a good question, like being concise, interesting, challenging, and purposeful. Those are senior requirements, which I believe are hard to evaluate. In this project, I will mainly focus on the basic 3 aspects.

Survey

Generally, this is quite an open task where no annotated datasets are provided. Besides, unlike common QG task on the news or Wikipedia entries, this task focus on evaluating questions from stories. The questions and answers must have strong connections with stories.

Commonly used referenced metrics like BLEU, ROUGE and METEOR, and Hybrid metrics do not work in this task due to the lack of annotation. As summarized in several papers[1][5], such referenced metrics counting word overlaps correlate poorly with human judgments when evaluating open-ended text. So I focus on exploring reference-free metrics. This paper[3] introduces the basic idea of *answerability*, which inspire me to use QA model to measure that. The PAQ paper[2] proposed "filtering" methods, including *local filtering* and *global filtering*. The basic idea of *local filtering* is also to measure *answerability*. It uses QA model to answer the generated questions, then judges whether the answer matches the original generated answer. This is helpful to remove bad questions. However, local filtering will not remove questions that are ambiguous, and can only be answered correctly with access to the source passage. The basic idea of global filtering is to use a ODQA model(DPR retriever + QA model) to answer questions without source passage. Unfortunately, this does not apply to this project, because the context of a story may be fictional, leaving a rare chance to retrieve related information from factual corpus like Wikipedia or news reports. But it does enlighten me to evaluate the ambiguity through Natural Language Inference(NLI) models.

Besides, Dataset Cartography[4] may also be an effective method to identify bad questions. The key idea is that, during the training procedure, questions that a model can confidently answer from the very first epoch are likely to be "easy", while questions whose predictions flip back and forth are likely to be flawed in some way.(cited from tryout project 2021) However, this method logically overlaps with my methods to some degree, so I currently do not adopt this idea.

Methods

I design methods to evaluate questions from 3 aspects.

Evaluate grammar and fluency

The best way to eliminate grammar and fluency errors is through a Grammar Error Correction(GEC) Model. While GEC models can be used to correct text, it's also powerful in grammar error detection. I adopted the most advanced model created by Grammarly, modifying its code in order to count the number of errors in each question. Then, I designed a metric called "grammar_score", which reflects the grammar and fluency level of questions:

$$Grammar\_score = \min(\max(1-(number\_of\_errors-1)/3, 0), 1)$$

Evaluate answerability

Inspired by PAQ method, I use a popular QA model on huggingface named "deepset/roberta-base-squad2". After feeding the contexts and questions into the QA model, I get the answers from it and compare them with original answers. The result is referred to as *answerability score*. This is similar to "local filter" proposed in PAQ paper. The QA model also gives a score indicating the confidence of such answers. According to my observation in case study, this score has a strong correlation with the accuracy of answers. So I use it as a part of my metric. Besides, the answer given by QA model may be longer and more detailed than the original answer, or the opposite. In most cases, both kinds of answers are appropriate. So I also take it into account in addition to exact match. Finally, the calculation of *answerability_score* is:

$$answerability\_score = answer\_score \times 0.5 + span\_match \times 0.5$$

where:

$$span\_match = \begin{cases} 1.0 & exact\ match \\ 0.8 & inclusion\ relation \\ 0 & otherwise \end{cases}$$

Evaluate inference relation

This requirement is derived from observations where many QA models can correctly answer questions even though the answer doesn't make sense. The possible reason behind this is that model can extract answers from the structure of the questions and contexts. Such tricks should not be encouraged when evaluating questions. To evaluate inference relations between contexts and QA-pairs, I firstly use a state of art QA2D model trained by myself(You can see more details from [here][https://github.com/SoyMark/Generating-Declarative-Statements-from-QA-Pairs]) to generate declarative answer sentences for each QA-pairs. Then I make use of a state of art NLI model named "roberta-large-mnli" to judge whether answer sentences can be deduced from contexts. The NLI model gives confidence scores on 3 different labels: contradiction, neutral and entailment. Specifically, the *inference_score* are defined as:

$$inference\_socre = \max(0, entailment\_score + 0.2 * neutral\_score - contradiction\_score)$$

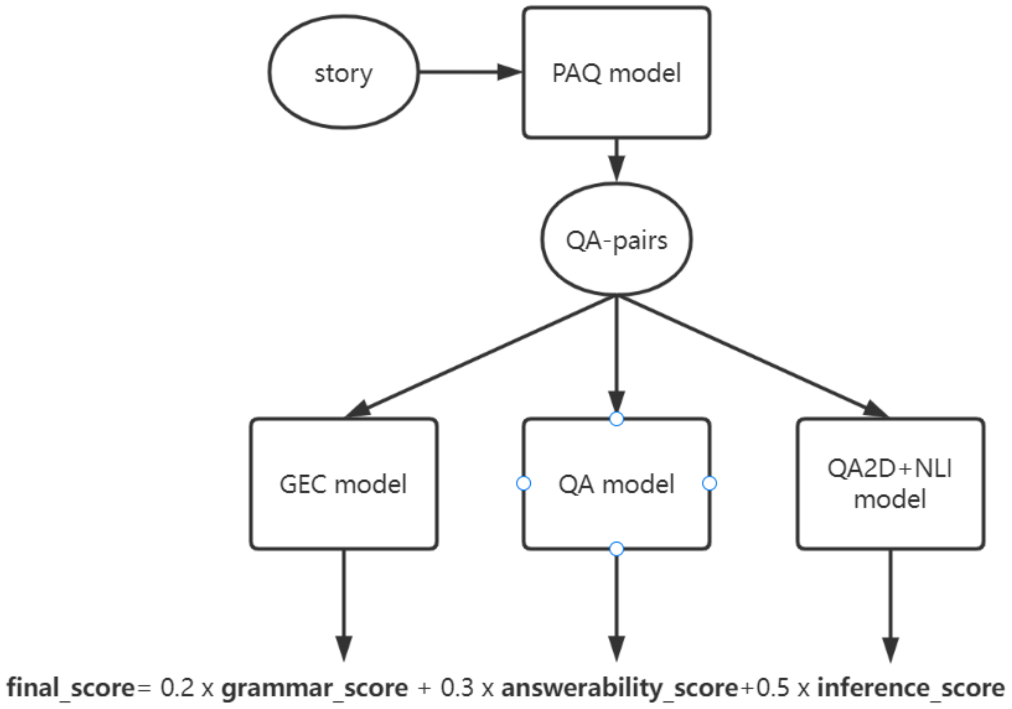
Generate final_score

Then, with three metrics that evaluate the generated question from different aspects, I decide to combine them with the weighted sum method:

$$final\_score = 0.2 \times grammar\_score + 0.3 \times answerability\_score + 0.5 \times inference\_score$$

The current weights are set by manual observation. As is discussed in Future work section, the weights can be adjusted through linear regression method, if I can find a dataset where questions generated from texts are evaluated by humans with a score. In practical situations, the weights can also be set according to specific needs.

The following flow chart shows the procedure of my methods.



story
PAQ model
QA-pairs
GEC model
QA model
QA2D+NLI model
final_score= 0.2 x grammar_score + 0.3 x answerability_score+0.5 x inference_score

Experiment and case study

To automatically generate Question-Answer pairs(QA-pairs) from a story, I used a part of PAQ model released on github. Specifically, I adopted the Answer Extraction component to extract entities that were likely to be asked in the question, and then adopted the Question Generation component to generate questions from the passage and answers. Then, I handled those questions according to Method section.

case study

I conducted experiments on several short stories. Here I list two of them. You can find the full stories in Appendix A.

The first case is questions generated from `The Fat Queen`. Questions are listed below in decreasing order of *final_score.* The higher the score, the better the question.

```
1. {'final_score': '0.976', 'answerability_score': '0.957',
'inference_score': '0.978', 'grammar_score': '1.000',
'question': 'what was the name of the fat queen', 'answer': 'Alina'}

2. {'final_score': '0.891', 'answerability_score': '0.656',
'inference_score': '0.988', 'grammar_score': '1.000',
'question': 'why did the fat queen become so fat', 'answer': 'because of an
unknown disease'}

3. {'final_score': '0.681', 'answerability_score': '0.501',
'inference_score': '0.662', 'grammar_score': '1.000',
'question': 'what happened to queen alina when she got sick', 'answer':
'very fat'}

4. {'final_score': '0.487', 'answerability_score': '0.849',
'inference_score': '0.198', 'grammar_score': '0.667',
'question': 'which king ordered everyone to eat more and look fat so that no
one would make fun', 'answer': 'King John'}

5. {'final_score': '0.443', 'answerability_score': '0.809',
'inference_score': '0.000', 'grammar_score': '1.000',
'question': 'how long did it take for the fat queen to stop eating',
'answer': 'a year and a half'}

6. {'final_score': '0.443', 'answerability_score': '0.809',
'inference_score': '0.000', 'grammar_score': '1.000',
'question': 'how long did it take for the fat queen to stop eating',
'answer': 'a year and a half'}

7. {'final_score': '0.365', 'answerability_score': '0.000',
'inference_score': '0.331', 'grammar_score': '1.000',
'question': 'what does the name of the fat queen mean', 'answer': 'fat'}

8. {'final_score': '0.200', 'answerability_score': '0.000',
'inference_score': '0.000', 'grammar_score': '1.000',
'question': 'what did the people of the kingdom call the fat queen',
'answer': 'fat and overweight'}
```

As you can see, the top 3 questions are natural and correct. The 4th question is also logically natural. However, the question is not complete, which results in a relative low *grammar_score*. The *inference_score* also suffers. I guess the reason behind this is that it's too long and incomplete. The 5th question is fluent and natural. However, the answer to the question obeys the meaning of the story. The queen never stops eating for a year and a half. As a result, the *inference_score* is 0. The 6th question is the same as the 5th. As for the 7th question, the answer repeats the information in the question, which is not likely to be a correct answer. This can explain the 0-point *answerability_score*. Finally, the 8th question has the wrong answer, resulting in a 0-point *inference_score* and *answerability_score*.

Here is another example. The ranked questions are generated from *Mysterious Stranger*.

```
1. {'final_score': '0.977', 'answerability_score': '0.988',
'inference_score': '0.962', 'grammar_score': '1.000',
'question': 'what kind of drink did the mysterious stranger drink',
'answer': 'lemonade'}

2. {'final_score': '0.839', 'answerability_score': '0.711',
'inference_score': '0.851', 'grammar_score': '1.000',
'question': 'what did the mysterious stranger put in his tote bag',
'answer': 'sunscreen'}

3. {'final_score': '0.821', 'answerability_score': '0.636',
'inference_score': '0.861', 'grammar_score': '1.000',
'question': 'what does the mysterious stranger look like', 'answer':
'tourist'}
```

```
4. {'final_score': '0.750', 'answerability_score': '0.562',
'inference_score': '0.763', 'grammar_score': '1.000',
'question': 'what was the purpose of the mysterious stranger', 'answer':
'rob a jewelry store'}

5. {'final_score': '0.631', 'answerability_score': '0.653',
'inference_score': '0.471', 'grammar_score': '1.000',
'question': 'what kind of chowder did the mysterious stranger bring',
'answer': 'clam chowder'}

6. {'final_score': '0.449', 'answerability_score': '0.518',
'inference_score': '0.187', 'grammar_score': '1.000',
'question': 'where did the mysterious stranger meet me', 'answer': 'shady
dance club'}

7. {'final_score': '0.261', 'answerability_score': '0.000',
'inference_score': '0.122', 'grammar_score': '1.000',
'question': 'where did the mysterious stranger get his money from',
'answer': 'The car accident'}

8. {'final_score': '0.261', 'answerability_score': '0.000',
'inference_score': '0.122', 'grammar_score': '1.000',
'question': 'where did the mysterious stranger get his money from',
'answer': 'The car accident'}
```

The top 3 questions are of good quality. The 4th one is a bit ambiguous since it doesn't tell the time. The answer to the 5th question is incorrect because it's the waiter who brought the clam chowder. The last three questions also have wrong answers which even contradict the story plot.

Conclusion

In this project, I defined good questions from 3 dimensions: 1) grammar and fluency; 2) answerability; 3) deductibility. Then, I did a comprehensive survey on reference-free metrics and methods of evaluating questions automatically generated from a story by QG models. Inspired by many ideas found in papers, I design methods to assess each dimension of the questions using advanced models. Finally, I did experiments and case studies on several stories, based on which I adjusted the weights of each assessment score and derived the final evaluation formula. According to manual observation of the results of ten selected stories, my methods had a promising performance in evaluating the quality of questions.

Future work

The current weights of each part assessment are determined by observation of several questions, which are not necessarily the best. As I mentioned in Method section, linear regression can be applied to determine the best weights, if there exists a dataset. However, I can barely find human-annotated datasets on questions generated automatically, making it hard to carry out the method. In the future, I will make a deeper investigation, or make a dataset myself to get better weights.

Appendix A

The full story mentioned in case study are as follow.

Mysterious Stranger

He really did look like a tourist, with a camera around his neck and a bottle of sunscreen sticking out of his tote bag. The portly man sat on the terrace, sipping lemonade and pretending to look at a glossy cruise brochure. His sunglasses masked his eyes, but I knew he wasn't looking at the brochure: he hadn't turned a page for the last ten minutes. As I brought him his clam chowder, he coughed up a "thank you" and looked at me briefly. I tried not to stare at the tiny scar across his left eyebrow. I walked back inside with my empty tray, shaking my head. He looked familiar, but I couldn't quite place him. Then it hit me. The car accident. The mysterious stranger who helped me out of my smashed car, just before it exploded. I rushed back to his table. He was gone. I moved his saucer and found his tip, along with a card: "I am deeply indebted to you. The night of your car accident, I was on my way to rob a jewelry store. Saving your life brought things back in perspective. I now live an honest life, thanks to you. God bless you! Mr. D."I shivered. The night of my car accident, I was heading for an interview in a shady dance club. Seeing human kindness through his heroic gesture turned my life around and brought faith back into my life. I unfolded the tip he left. Among the singles was a grand with a pen mark underlining "In God We Trust."I said a silent prayer for him and got back to work, smiling. (275 words)

The Fat Queen

Once upon a time, there was a queen named Alina, who was the fattest woman in the whole kingdom.Queen Alina was normal before, but because of an unknown disease her weight rapidly increased and she became very fat.The queen tried every possible way to cure her disease, but she had failed.Now some people of the kingdom had started making fun of the queen, behind her back they used to call her the fat and ugly queen Alina. Wherever queen Alina used to go anywhere, everyone stared at her with surprise and like they were watching a creature.When king John got to know the behavior of his people toward her queen, he got very angry and ordered every person in the kingdom to eat more and look fat so that no one will make fun of her.The queen tried to stop the king's order, but he listened to no one.King John even ordered his guards to throw everyone outside the kingdom who was not following his orders.Afraid of the king, most of the people, including children started eating more. For those people who were poor and couldn't afford to eat more, the guards of the kingdom used to provide extra food to eat.Every month, some guards used measured the weight of every person in the kingdom, and who were not found increasing their weight, the guards used to destroy their houses and displace them from the kingdom. After a year and a half, everyone in the kingdom, including children was fat and overweight.But now the teenager turned princess Bella had started feeling abnormal and odd as whomever she saw in the kingdom looked fat and even her all her friends were so chubby and overweight but she was not.The princess said to her mother that she also wants to look like everyone else.Her mother told her that it was a punishment given by her father to the people of the kingdom, not for us.The princess replied that now all her friends make fun of her as she doesn't look like them.The queen didn't know what to do for her daughter.Soon depressed Bella also started eating more to look like everyone else and like her friends.The king and queen tried everything to stop her, but the princess never listened to them and one day she got very ill because of her new overeating habit.Looking at his own sick and overweight daughter the king realized his great mistake. Finally, the king revoked his order and asked everyone to stop overeating.The queen apologized on behalf of the king to her people and said that just because some people made fun of her obesity, the king punished the whole kingdom.The people of the kingdom also apologized for their mistake and said now they know how it feels to be fat and overweight and asked the king and queen to forgive them.After a few months, some became successful in turning themselves into what they looked like before, but for most people, it was so hard to leave their new overeating habit. (505 words)

Reference

[1] Amidei J, Piwek P, Willis A. Evaluation methodologies in automatic question generation 2013-2018[J]. 2018.

[2] Lewis P, Wu Y, Liu L, et al. Paq: 65 million probably-asked questions and what you can do with them[J]. Transactions of the Association for Computational Linguistics, 2021, 9: 1098-1115.

[3] Nema P, Khapra M M. Towards a better metric for evaluating question generation systems[J]. arXiv preprint arXiv:1808.10192, 2018.

[4] Swayamdipta S, Schwartz R, Lourie N, et al. Dataset cartography: Mapping and diagnosing datasets with training dynamics[J]. arXiv preprint arXiv:2009.10795, 2020.

[5] Guan J, Huang M. UNION: an unreferenced metric for evaluating open-ended story generation[J]. arXiv preprint arXiv:2009.07602, 2020.