Evaluating BERT-based Rewards for Question Generation with Reinforcement Learning

Peide Zhu
Delft University of Technology
Delft, The Netherlands
P.Zhu-1@tudelft.nl

Claudia Hauff
Delft University of Technology
Delft, The Netherlands
c.hauff@tudelft.nl

ABSTRACT

Question generation systems aim to generate natural language questions that are relevant to a given piece of text, and can usually be answered by just considering this text. Prior works have identified a range of shortcomings (including semantic drift and exposure bias) and thus have turned to the reinforcement learning paradigm to improve the effectiveness of question generation. As part of it, different *reward functions* have been proposed. As typically these reward functions have been empirically investigated in different experimental settings (different datasets, models and parameters) we lack a common framework to fairly compare them. In this paper, we first categorize existing rewards systematically. We then provide such a fair empirical evaluation of different reward functions (including three we propose here) in a common framework. We find rewards that model *answerability* to be the most effective.

CCS CONCEPTS

• **Information systems** → *Question answering*.

KEYWORDS

Question generation, reinforcement learning, reward functions

ACM Reference Format:
Peide Zhu and Claudia Hauff. 2021. Evaluating BERT-based Rewards for Question Generation with Reinforcement Learning . In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Question generation (QG) systems aim to generate natural language questions that are relevant to a given piece of text (the so-called *context*—typically a sentence or a paragraph), and can usually be answered by just considering the context. As an important natural language processing task, QG can be used to improve question-answering [11, 58], conversational systems [48], and information retrieval (IR) [55, 57]. As a concrete example of the latter, QG has been employed to improve the retrieval effectiveness of search systems by expanding documents with generated questions that the document might answer [35, 36]. The use of automatic QG has also recently been shown to be beneficial for learners in an interactive reading experiment [45], aiding learners' comprehension and learning. The natural next step is to employ question generation in the *search as learning* area [3], which consists of interactive reading, searching and browsing activities [9, 27, 46, 49, 60].

The current state-of-the-art QG systems are based on deep neural networks, which input the context (as well as—in many approaches—the *answer* to the to-be-generated question) into an encoder, and generate a question about the context (and the provided answer) with a decoder.

Many datasets have been employed for QG research, such as `SQuAD` [40, 41], `MS MARCO` [34] and `HotpotQA` [52]. In these datasets, only one ground-truth question is provided for each question-answer pair. However, for each context paragraph, there are usually several different facts related to the answer that questions can be generated about. In addition, even if there is only one fact contained in the answer, several syntactically very different questions may semantically be strongly related or even the same.

Based on these two observations, it is clear that the ground-truth questions provided in these datasets are not sufficient for high-quality question generation purposes. In fact, prior works have found that the likelihood-based training suffers from the problem of *exposure bias* [42], i.e., the model does not learn how to distribute probability mass over sequences that are valid but different from the ground truth. Because of exposure bias, many QG models are not trained well enough to discover the relations between context and questions. In addition, QG models trained in this manner can also suffer from the *semantic drift* problem, i.e., the models ask questions that are not relevant to the context and answer [58].

As a response to these training regime and dataset shortcomings, recently the reinforcement learning (RL) paradigm has been taken up by the research community in order to optimize the QG model during training with *rewards* that can directly evaluate question quality next to the available likelihood-based loss, so that questions with different forms from the ground-truth can be explored [1, 12, 18, 54, 58].

In the literature, a number of very different types of rewards have been proposed to evaluate question quality automatically such as the n-gram based metrics BLEU, Meteor and Rouge [1, 20, 44], the answerability reward [56, 58], and fluency [50, 56]. However, as reported by Hosking and Riedel [19] high RL-based rewards do not always equate to better questions when evaluated in a human evaluation setting. Though undoubtedly, achieving a high score in a

This research has been partially supported by *NWO* projects *SearchX* (639.022.722) & *Aspasia* (015.013.027) as well as the China Scholarships Council (No. 201906340170).

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Three examples of automatically *Generated* question, the *Context*, the ground-truth answer span, the question was generated for and the *Ground truth* question. The numeric columns represent the exact n-gram match metrics (BLEU-4), heuristic n-gram based metrics (Meteor), answerability (BERT-QA-loss), semantics-based similarity (QPP) and relevance based rewards (C-Rel, CA-Rel). These rewards are explained in more detail in Section 3.3. The scores range from 0 to 100.

Example 1		**B-4**	**Meteor**	**BERT-QA-loss**	**QPP**	**C-Rel**	**CA-Rel**
Context	At the end of World War I, the Rhineland was subject to the Treaty of Versailles.						
Ground truth	When was rhineland subject to the treaty of versailles ?						
Generated	The treaty of versailles was subject to the treaty of versailles?	53.32	77.82	0.02	1e-5	5e-5	0.87
Example 2							
Context	The clinical pharmacist's role involves creating a comprehensive drug therapy plan for patient-specific problems, identifying goals of therapy, and reviewing all prescribed medications prior to dispensing and administration to the patient. The review process often involves an evaluation of the appropriateness of the drug therapy (e.g., drug choice, dose, route, frequency, and duration of therapy) and its efficacy.						
Ground truth	What is involved in a review of prescribed medications?						
Generated	What does the review process often use?	0	14.31	100	99.94	99.98	99.91
	Example generated questions with issues (Section 4.2); assigned rating is shown in (brackets)						
Syntax issue (2)	what does the review process often involves ?	0	16.85	100	99.48	99.86	98.47
Non-answerable (0)	who does the review process involve ?	0	11.36	0.53	97.7	99.9	100
Relevance issue (1)	what is the dose of the drug ?	0	28.71	0.11	0	99.97	99.92
Example 3							
Context	This is the most common method of construction procurement and is well established and recognized. In this arrangement, the architect or engineer acts as the project coordinator.						
Ground truth	In the most common construction procurement, who acts as the project coordinator ?						
Generated	Who is the project coordinator?	14.16	36.6	100	97.83	99.98	99.98

human evaluation is more important than an automatic evaluation metric.

Our work aims at further improving QG as the existing rewards are not sufficient. We motivate this argument by the following three observations about the datasets, setup of the task and evaluation metrics. First, n-gram based metrics evaluate question quality by computing the exact match of n-grams in the generated and ground-truth questions. On the one hand, these metrics may give high scores for low-quality generated questions which repeat *n*-grams (such as shown in Example 1 in Table 1 where the term *versailles* appears twice in the generated question); on the other hand, since multiple questions are valid but only one ground-truth question is provided, these metrics can also fail to appropriately score question paraphrases and semantically equivalent questions (as shown in Examples 2 and 3 in Table 1). Second, there are several essential components involved in the generation and evaluation of a question: *the context*, *the answer* and *the ground-truth*. However, most of the proposed automatic metrics only consider one of them. For example, the n-gram based metrics compare the generated question with the ground-truth question; the answerability metric evaluates whether the question can be answered given the context, and the fluency metric computes the perplexity of the generated question.

Therefore, we argue that further work is required to investigate how to *jointly* use these three components.

Lastly we point out that previously introduced rewards have been empirically investigated in different experimental settings (datasets, model, parameters), which does not enable us to compare their effectiveness directly.

We make two contributions in our work: first of all, we propose three novel rewards; secondly, we provide a thorough empirical evaluation of the previously introduced rewards employed inside a common base model. This in turn allows us to compare the impact different rewards have on the model quality. Concretely, we decided to use BERT [6] (due to its strong performance across a wide range of NLP tasks) as the base model to provide rewards for QG.

Overall, our main finding is that in such a fair comparison the rewards that model *answerability* are the most effective, both in terms of an automatic evaluation as well as a human evaluation.

2 BACKGROUND

In this section, we first discussion question generation, then turn to common evaluation metrics used to evaluate QG approaches.

2.1 Question Generation

As an important natural language processing task, QG has a wide range of applications; we here discuss three types of applications (QA, conversational systems and human learning) and then discuss the types of QA generation approaches that exist.

2.1.1 QG for QA. As the available information online and the requirement of quick access to information grows, question answering (QA) is playing an ever more important role. As a dual task of question-answering, QG can be used to improve QA performance. Some works [7, 11, 26, 46, 58] take QG as a generator to harvest question-answer pairs from passages, and use this harvested data to pre-train QA models, which subsequently resulted in improved QA model effectiveness. QG is also widely used in IR tasks, such

as improving search system effectiveness by generating clarifying questions [57], or generating questions from e-commercial customers reviews [55].

2.1.2 QG for Conversational Systems. Conversational systems have become an important tool for information seeking. Asking good questions is significant for both providing user interaction, and for conversational QA training. Yao et al. [53] used QG to create conversational characters. Wang et al. [48] and Ling et al. [25] proposed learning to ask questions in open-domain conversational systems with conversational context information. Gao et al. [14] and Gu et al. [15] proposed to use conversational question generation and conversation flow modeling as a means to generate synthetic conversations for training and evaluation purposes.

2.1.3 QG for Learning. Questions are a fundamental tool for a variety of educational purposes. Manual construction of good learning-oriented questions is a complex process that requires experience, resources and time. To reduce the expenses of manual construction of questions and satisfy the need for a continuous supply of new questions, QG techniques are introduced. Kurdi et al. [21] provide a systematic review of QG works for educational purposes. Besides, by conducting an interactive reading experiment and gaze tracking, Syed et al. [45] showed that the use of automatic QG is indeed beneficial for learners as it aids learners' comprehension and learning.

2.1.4 QG Approaches. Past question generation research can be categorized as rule-based and neural network based on the generation approach employed. The rule-based approaches [16, 17, 24, 30–32] rely on well-designed manually created templates and heuristic linguistic and semantic rules for question generation. Labutov et al. [22] proposed a pipeline for question templates generation by crowdsourcing and ranking. Other works [4, 11, 28] proposed to generate factoid source question-answer triplets from passages, subtitles, or wiki knowledge graphs. Inspired by the advances in applying deep learning in natural language generation, various neural network models have been proposed for question generation [1, 1, 10, 27, 29, 46, 47, 49, 56, 60]. These models formulate the question generation task as a sequence-to-sequence (Seq2Seq) neural learning problem with different types of encoders, decoders and attention mechanisms.

2.2 RL-based Question Generation

To address the exposure bias and *semantic drift* problem, the reinforcement learning (RL) paradigm has been taken up by the research community in order to optimize the QG model during training with *rewards* that can directly evaluate question quality next to the available likelihood-based loss, so that questions with different forms from the ground-truth can be explored. Rennie et al. [43] proposed an effective efficient optimization approach called self-critical sequence training (SCST). SCST utilizes its own test-time inference algorithm output to normalize the rewards it experiences. Estimating the reward signal and estimating normalization is avoided, while at the same time harmonizing the model with respect to its test-time inference procedure. Because of its effectiveness, SCST is commonly used in follow-up RL-based QG methods, while they use different evaluation metrics calculated with different methods and models as rewards [1, 12, 18, 54, 58]. We will discuss these metrics in the next section.

2.3 QG Evaluation Metrics

As a natural language text generation task, most previous QG works use traditional metrics such as BLEU and Rouge to evaluate generated questions by comparing them with the ground-truth questions. However, Novikova et al. [37] and Nema and Khapra [33] pointed out that human ratings about question quality or answerability do not correlate well with these automatic evaluation metrics. Therefore, several different metrics have been proposed to evaluate different aspects of question quality, including fluency [50, 56], answerability [33, 50, 58], paraphrasing [19, 58], or discriminator-based relevance [50]. We broadly categorize question evaluation metrics used in prior works into $n-$gram-based and learned metrics, based on the underlying methods they use.

2.3.1 $n-$Gram-based Metrics. BLEU [38] is the most widely used metric in machine translation and QG. It is computed by how much the $n-$gram ($n = 1, 2, 3, 4$) in the predictions (here: the *generated question*) can be matched in the reference (here: the *ground-truth question*). Meteor [5] computes not only the exact unigram match precision and recall, but also allows the matching of word stems, synonyms, and paraphrases. Meteor also put weights on different content and matching types. Rouge$-n$ ($n = 1, 2$) [23] computes the recall rate of $n-$grams of the reference and the predictions, while Rouge$-L$ is a variant of Rouge-1, but uses the length of the longest common subsequence to compute the match rate.

2.3.2 Learned Metrics. As learned word embeddings [39] or contextual embeddings [6] have been shown to provide better representations for capturing the lexical and semantic similarity, various metrics have been proposed that use these learned embedding or neural models to optimize the correlation with human judgments, such as SMS [2] or BERTscore [59]. Different from general text evaluation metrics, question quality evaluation requires a special focus on answerability, relevance to both context and the ground-truth. Many metrics have been proposed to fulfill these special requirements, such as Q-metrics [33], QPP and QAP [58], or question-specific rewards [50].

3 METHODOLOGY

We now present our methodology. In order to evaluate the different rewards, we designed a common framework that provides a fair testbed. This framework is visualized in Figure 1. It consists of two parts: the *QG model* and the *reward evaluator*. We see that beyond the reward computations (which are described in detail in this section), the remainder of the framework is the same, no matter the reward employed.

Generally, we use C and $\mathbf{A}$ to represent the context, and answer span respectively. Here, the context is comprised of a sequence of words $C = [w_i]_{i=1}^{M}$ with M being the size of the context. The answer span $\mathbf{A} = \{A^s, A^e\}$ indicates the start and end position of the answer in the context. Let $\hat{Q}$ represent the generated question, which is a sequence of predicted tokens $\hat{Q} = y_0, y_1, \ldots, y_N$. Then,

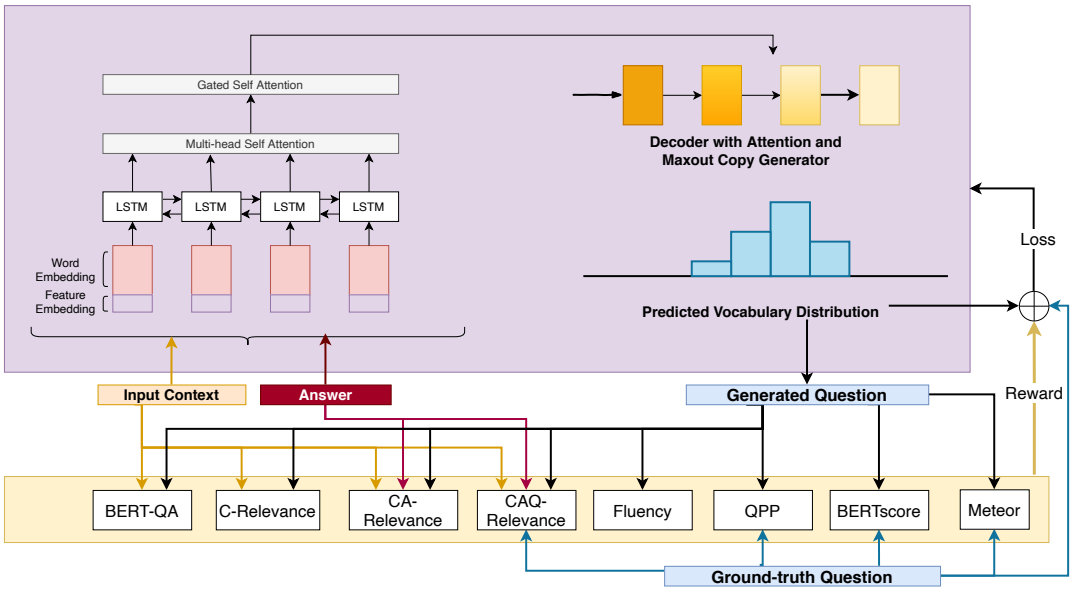

Figure 1: Architecture of our proposed question generation model.

the question generation task can be formalized as:

$$\hat{Q} = \arg\max_{Q} P(Q|C, A)$$

We now describe our two framework components (QG model and reward evaluator) in turn, before detailing the different rewards we implemented.

3.1 Question Generation Model

The QG model uses the Seq2Seq framework with a maxout pointer mechanism and gated self-attention network similar to Zhao et al. [60] for paragraph-level question generation, as it is straightforward, and similar models have been widely employed in recent QG research. To utilize the long distance relation information at paragraph-level we add a multi-head attention mechanism in the encoder. We use the unsupervised pre-trained Glove [39] embeddings to initialize our word embeddings, as Glove embeddings have learned the substructure and statistical relation among words. In terms of word embeddings, besides word vectors we also include word feature embeddings, including the part-of-speech (*POS*), named entity (*NE*) and answer tag. The answer tag vector is used to indicate whether a word is in the answer span. The *POS* and *NE* labels were extracted with Spacy[1].

3.2 Reward Evaluator

We use the self-critical sequence training (SCST) algorithm [43] for RL. SCST is an efficient reinforcement algorithm that directly utilizes the test-time inference output to normalize the rewards it experiences. In this setting, the evaluators are the *environment* and the QG model is the *agent* that interacts with it. The QG model's parameters θ define a generation policy (i.e. the predicted token probability) $\mathbf{P}_\theta$ which makes the prediction of the next word, i.e. the *action*. After each action, the agent updates its state, i.e. updates hidden states, weights, etc. of the QG model. Once the agent finishes generating a sequence Q, it observes a *reward* $r(Q)$ which is posed by evaluators, computed by comparing it to the corresponding ground-truth sequence Q^* with a given reward metric. Then the RL loss function is defined as:

$$\mathbf{L}_{rl} = -\mathbf{E}_{Q^s \sim \mathbf{P}_\theta}(r(Q^s))$$

where Q^s is the sampled output produced by multinomial sampling, that is, each word q_t^s is sampled according to the likelihood $P(q_t|X, q_{<t})$ predicted by the generator. Because the sampling procedure is non-differentiable, the policy gradient $\nabla_\theta \mathbf{L}_{rl}$ is approximated using the baseline output Q^b obtained by greedy search, that is, by maximizing the output probability distribution at each decoding step. The loss function, when instantiated as just discussed, becomes thus:

$$\mathbf{L}_{rl} = (r(Q^b) - r(Q^s)) \sum_t \log \mathbf{P}(q_t^s|X, q_{<t}^s).$$

Using this reinforcement loss alone does not result in correctly learnt word probabilities. For this reason, we follow the mixed objective approach [1], combining both cross-entropy loss (base model loss) and the RL loss:

$$\mathbf{L}_{mixed} = \lambda \mathbf{L}_{rl} + (1 - \lambda)\mathbf{L}_{base}.$$

Here, λ is a mixing ratio to control the balancing between RL loss and the base model loss. In the following sections, we will explain the rewards in detail.

3.3 Rewards

We categorized the rewards from the literature into different reward *types* as shown in Table 2. Importantly, in Table 2 we also provide

[1]https://spacy.io/usage/linguistic-features

insights into what information (context, answer, ground truth question, generated question) the reward functions take as input. Naturally, all rewards take the generated question into account, however beyond that there is little agreement as to what else to use. Based on the inputs to the reward function, and the downstream model type, we categorize the rewards into four types: (i) **fluency** indicates whether the generated question is a valid expression according to the language model; (ii) **similarity** indicates the similarity between the generated question and the ground-truth question; (iii) **answerability** indicates whether the generated question can be answered given the context; and (iv) **relevance** indicates how the generated question is relevant to the context, or the combination of the context, the answer and the ground truth.

The *BERT-Task* in Table 2 is the downstream task of BERT we use to compute the rewards. For the Fluency and BERTscore rewards, we use the contextual embeddings of BERT as the language model. For discriminators, we add the BERT model transformer with a sequence classification head on top of the pooled output as classifier. For the QA task, we use the BERT model with a span classification head on top to predict the start and end positions of the answers.

Lastly we point out that we indicate in Table 2 also the three novel reward functions we contribute in our work: BERTscore, CA-Rel and CAQ-Rel.

Table 2: List of categorized reward functions employed in our work. Shown here are the inputs used to compute each reward. *GT* refers to the ground truth and *GQ* refers to the generated question. The novel rewards for QG we propose in this work are labeled with ⋆.

Reward	BERT Task	Context	Answer	GT	GQ
Fluency category					
Fluency [50]	LM				✓
Similarity category					
⋆ BERTscore	LM			✓	✓
QPP [58]	Classifier			✓	✓
Answerability category					
BERT-QA-loss[58]	QA	✓	✓		✓
BERT-QA-geo[50]	QA	✓			✓
Relevance category					
C-Rel [50]	Classifier	✓			✓
⋆ CA-Rel	Classifier	✓	✓		✓
⋆ CAQ-Rel	Classifier	✓	✓	✓	✓

We now discuss the different reward functions in the order of their appearance in Table 2.

3.3.1 Fluency Category. The perplexity of a sentence under a well-trained language model usually serves as a good indicator of its fluency [51]. We adopt the LM-based fluency reward as proposed by Xie et al. [50]. We first fine-tune the BERT language model with questions from the SQuAD dataset. The fluency reward R_{flu} for question Q is calculated as follows:

$$R_{flu} = -\exp\left(-\frac{1}{|Q|}\sum_{i=1}^{|Q|}\log M_{flu}(Q_i|Q_{<i})\right)$$

3.3.2 Similarity Category. The n-gram based automatic evaluation metrics (BLEU, Meteor and Rouge) score the question similarity by computing the exact match of n-grams in the generated and ground-truth questions. As pointed out in Section 1, these metrics may yield a high score for low-quality generations which repeat n-grams in the generated question sequence. As there may be many valid questions with similar semantics, but only one ground truth question is provided, these metrics can also fail to appropriately score question paraphrases and semantically similar but syntactically very different questions. Therefore, we investigate two semantics-based question similarity rewards: *BERTscore* (the use as reward for QG we propose) and *Question Paraphrasing Probability (QPP)*. These two rewards are based on BERT, and compute the semantic similarity with high-level contextual representations instead of exact or heuristic n-gram matching.

BERTscore-based Reward. BERTscore [59] scores the similarity between the generated question (the *generation*) and the ground-truth question (the *reference*) by computing a similarity score for each token in the generation with each token in the reference. In contrast to n-gram-based metrics, BERTscore first represents contextualized token vectors with BERT and then uses greedy matching to maximize the matching similarity score, where each token is matched to the most similar token in the other sentence; subsequently precision and recall are computed to yield the F1 measure. Given the generated question $\hat{Q}$ and the ground-truth question Q, the BERTscore can be computed as follows:

$$R_{BERT} = \frac{1}{|Q|}\sum_{y_i\in Q}\max_{\hat{y}_j\in\hat{Q}}\mathbf{y}_i^T\hat{\mathbf{y}}_j$$

$$P_{BERT} = \frac{1}{|\hat{Q}|}\sum_{\hat{y}_i\in\hat{Q}}\max_{y_j\in Q}\hat{\mathbf{y}}_i^T\mathbf{y}_j,$$

$$F_{BERT} = 2\frac{P_{BERT}\dot{R}_{BERT}}{P_{BERT}+R_{BERT}}.$$

Here, y_i is the i^{th} token in question sequence, and $\mathbf{y}_i$ is the pre-normalized contextual vector generated by BERT. We use the F_{BERT} score as our reward.

QPP-based Reward. Given one reference, n-grams based metrics sometimes fail to evaluate question paraphrases appropriately. Thus, inspired by the QPP reward proposed by Zhang and Bansal [58], we propose a BERT-based question paraphrasing classifier to provide paraphrasing probability as a reward. We pre-train this classifier model with the Quora Question Pairs dataset[2]. As shown in Example 2 of Table 1, it scores question paraphrases more fairly: given the ground-truth question *What is involved in a review of prescribed medications?* and the generated question *What does the review process often use?*, we find BLEU-4 to assign these semantically similar questions a score of 0 while QPP assigns a score of 99.94. During the training of the QG model, we use the QPP

[2]https://www.kaggle.com/c/quora-question-pairs

classifier to provide the probability of the generated questions and the ground-truth question being paraphrased as the reward.

3.3.3 Answerability Category. The answerability of a question evaluates whether the generated question can be answered given the context. There are several reasons to consider answerability as reward for QG. First, for many QG applications, such as generating questions for reading comprehension or question answering, it is a common requirement to ask questions that can be answered with the context information. Second, semantically drifted questions usually cannot be answered by the given context and answer, such as the the *relevance issue* and the *non-answerable* question shown in Example2 in Table 1. Third, given the context, several valid questions are usually valid for the answer. Some contain information that is not used in the ground-truth. The question similarity based metrics cannot evaluate this kind of novel generation fairly. Besides the ground-truth question, the answerability reward can take the context information into consideration. Therefore, we investigate two BERT-QA based answerability rewards. One is based on the QA loss *(BERT-QA-loss)*, and one is a heuristic reward based on the geometric average of the QA probability *(BERT-QA-geo*). We use the BERT-QA model which is pre-trained on SQuAD to provide the QA probability, i.e. given the input context C, the ground-truth answer $A = \{A^s, A^e\}$ and the generated question $\hat{Q}$, the question answering model outputs two probability distributions $P^s_{ans} = P(A^s|C, \hat{Q})$ and $P^e_{ans} = P(A^e|C, \hat{Q})$ over tokens in C, where $P^s_{ans}(i)/P^e_{ans}(i)$ is the probability that the i-th token is the start and end position of potential answer spans in the context.

BERT-QA-loss Reward. Given the ground-truth answer $A = \{A^s, A^e\}$, we evaluate the answerability by computing the cross-entropy loss of the QA predictions with the ground-truth answer:

$$loss(P^s_{ans}, P^e_{ans}, A) = CE(P^s_{ans}, A^s) + CE(P^e_{ans}, A^e)$$

$$R_{ans}(C, Q, A) = e^{-loss}$$

.

BERT-QA-geo Reward. As argued by Xie et al. [50], when the question is answerable, the model should be quite confident about the start/end span of the answer, so the distribution should peak for both P^s_{ans} and P^e_{ans}, i.e., the value of $max_i P^s_{ans}(i)$ and $max_j P^e_{ans}(j)$ are both large. Therefore, the geometric average of these start and end position probability distributions can be used as a heuristic answerability reward:

$$R_{ans}(C, Q) = \max_{1 \le i \le j \le T, j-i \le l} \sqrt{P^s_{ans}(i|C, Q) \cdot P^e_{ans}(j|C, Q)}.$$

Here, l represents the maximum answer length.

3.3.4 Relevance Category. There are several essential components involved in the generation and evaluation of a question: *the context*, *the answer* and *the ground-truth*.

We investigate a series of binary classifier based discriminators to judge whether the generated question is relevant to the context *(C-Rel)*, the context and answer *(CA-Rel)*, and the context, answer and reference*(CAQ-Rel)*. While the first reward (C-Rel) stems from prior work, we extended it and propose the just mentioned two novel rewards for QG (which include more information than C-Rel in the input).

C-Rel Reward. This reward indicates whether a question is relevant to the context. We design a binary classifier based on BERT, inspired by Xie et al. [50]. It takes the context C and the generated question $\hat{Q}$ as inputs and the output is the probability that $\hat{Q}$ is relevant to C. To fine-tune the BERT classifier, we use the ground truth questions provided in the SQuAD dataset as the positive samples. We create negative samples in two ways: based on (i) question swapping and (ii) entity swapping. Negative sampling based on question swapping means to randomly select ground-truth questions about a different context C as negative question samples for context C. In contrast, negative sampling based on entity swapping means to replace entities in the ground truth question with entities that do not occur in the context. We prefer to select entities that are of the same entity types, such as locations, dates and names. Secondly, we create negative samples based on entity swapping by replacing entities in ground truth questions with the entities from the same context though of different entity types.

CA-Rel Reward. We propose to use the probability that $\hat{Q}$ is relevant to the context C and the answer A pair as reward. We design a BERT-based binary classifier which takes the context, the answer and the generated question as inputs.

As there is only one ground-truth question for each context-answer pair, it is a challenge to create enough positive samples to train the classifier. We use three approaches to create positive samples: (i) back translation, (ii) information from a large paraphrase database and (iii) a neural paraphrasing model. We now discuss each of these options in more detail. Paraphrases can be obtained by translating an English string into a foreign language and then back-translating it into English [8]. We select German as the pivot, and use two pre-trained neural translation models: English-German and German-English to generate question paraphrases. The PPDB [13] is a large-scale paraphrase database containing over a billion of paraphrase pairs in 24 different languages. In our work, we employ bidirectionally entailing rules from PPDB, which are replacing single word or phrases with their paraphrases in PPDB. Finally, we train a seq2seq translation model with the Quora Question Pairs dataset, and apply beam search to decode paraphrasing questions. Having created positive samples in these manners, we are left with creating negative samples for each question: we here employ the same manner as described for C-Rel.

CAQ-Rel Reward. Lastly, we propose a binary classifier which takes the context, answer, ground-truth Q_G and the generated question as input, and outputs the probability that the generated question is relevant to the triplet $\{C, A, Q_G\}$. We create the positive and negative samples in the same way as described for CA-Rel.

4 EXPERIMENTS

We conduct our experiments on the SQuAD 1.0 [41] dataset which is widely used in QG and QA research [10, 26, 57, 58]. It contains over 100K question-answer pairs generated by crowd-workers from 536 Wikipedia articles. The answers are selected word spans from Wikipedia article sentences. The dataset contains publicly accessible train and validation splits and a privately hosted test split. We split

the public validation set into two parts as the development set and the test set. Thus, we have 87,598/5,285/5,285 samples for training, validation and testing respectively.

In a first step, we train all the proposed rewards. We employ huggingface's[3] PyTorch BERT implementation in its uncased variant.

For the answerability rewards, we fine-tune BERT for the QA model with the SQuAD dataset. On the test set, the fine-tuned model obtains 80.28% exact match score and 87.89% F1 score.

For the fluency reward, we fine-tune the BERT language model with ground-truth questions in SQuAD and achieve 23.29 perplexity on the development set.

For BERTscore, we use the available BERTscore implementation[4] provided by Zhang et al. [59]. This model does not require further fine-tuning.

We use the BERT model with a linear layer on top of the pooled output as the discriminator for the QPP reward and all three rewards in the relevance category. We train the model for all rewards with different datasets. For the QPP reward, we rely on the Quora Question Pairs dataset, and spit the dataset as train/dev/test sets following the ratio of 70%, 15% , 15%, which expressed in numbers of samples amounts to 283K/60,643/60,643 respectively. For the C-Rel reward, based on the dataset creation strategy mentioned in Section 3.3.4, we harvest 297,980/17,322/17,954 samples for training, validation and testing respectively. For the CA-Rel reward, we harvest 1,137,052/68,649/68,703 samples as the training, development and test set. Finally, for the CAQ-Rel reward, the size of the training, development and test sets are 560,774/33,809/33,177. The performance of the trained models used as rewards is summarized in Table 3. Numbers are reported on each task's test set. In all cases, the accuracy reaches at least 90.98, indicating that our training regime yielded highly accurate models.

Table 3: Fine-tuned BERT-based classifier effectiveness.

Reward	Precision	Accuracy	Recall	F1
QPP	85.7	90.98	90.68	88.12
C-Rel	86.06	92.20	87.70	86.87
CA-Rel	93.27	92.62	92.99	93.13
CAQ-Rel	97.67	97.86	98.95	98.30

Before RL training with these rewards, we first train the basic QG model by minimizing the cross-entropy loss and the copying loss. The encoder of the basic QG model uses a 2 layer bi-directional LSTM. The LSTM hidden cell size is 300. A dropout layer with probability 0.3 is applied between two bi-directional LSTM layers. We keep the 30K most frequent words in SQuAD as vocabulary. The word embedding size is 300. The decoder uses a 1 layer LSTM. We use SGD with momentum for optimization (momentum value is 0.8). The initial learning rate is 0.1 and decreases linearly after half of training steps. We use beam search (beam size 10) for the decoding. We first train the basic QG model for 16 iterations, then we fine-tune the basic model with RL training, as described in Section 3.2. The mixing ratio (λ) in RL is set to 0.2. We use the basic QG model as our baseline to compare performance of all the rewards. To compare the BERT-based rewards with n-gram based metrics, we also train our QG model with a Meteor-based reward. We choose Meteor as the representation of n-gram based rewards as based on our previous experience, Meteor usually outperform other n-gram rewards.

[3]https://huggingface.co/transformers/
[4]https://github.com/tiiiger/bert_score

4.1 Automatic Evaluation

We investigate the QG models' performance along n-gram based automatic evaluation metrics and the proposed rewards. The automatic metrics we use are BLEU, Meteor and Rouge-L. They are based on the n-gram similarity between the generated questions and the ground truth, and are commonly used in text generation tasks. We calculate these metrics with the package released by Du et al. [10].

Table 4 summarizes our main results. We make the following key observations:

(1) Training the QG model with RL on every reward leads to better effectiveness with respect to the automatic metrics, except for the fluency reward on Meteor. **This result shows that it is effective to apply reinforcement learning on QG model training in terms of the automatic metrics.**
(2) Optimizing one reward always leads to the improvement of the corresponding reward score. But the improvement of each reward varies from each other, e.g. when optimizing the CAQ-Rel reward, the CAQ-Rel score improves by 5.02 compared to the baseline; however, optimizing the fluency reward only leads to a 0.02 improvement. This shows that the degrees of how rewards influence QG training differ.
(3) The rewards we use can be categorized into four types as already outlined in Section 3.3: fluency, answerability, similarity and relevance. **We find optimizing one reward also leads to a score increase for other rewards of the same type.** This implies that rewards of the same type are correlated. We further investigate the correlation between them. The correlation matrix (expressed in Pearson correlation coefficient) is shown in Figure 2. We find the similarity based rewards BERTscore and QPP are strongly correlated to each other, with the correlation coefficient 0.62. The relevance based rewards are more related to the similarity rewards than each other. The BERT-QA-loss reward and the BERT-QA-geo reward are almost independent, which shows the heuristic reward BERT-QA-geo may be not a good indicator whether a question can be answered by a QA model. This insight is useful for designing unsupervised QA system. The BERT-QA-loss reward and the fluency reward are not correlated to other rewards, which shows the fluency and BERT-QA-loss reward focus on different aspects of the generated questions' quality.

4.2 Human Evaluation

In addition to the automatic metrics, we further conduct a human evaluation on our test set to investigate whether optimizing the proposed rewards leads to the improvement in question quality by humans' standard.

Table 4: Performance evaluation along automatic metrics and rewards. The automatic metrics are BLEU-3 (B-3), BLEU-4 (B-4), Meteor (M) and Rouge-L (RGL).

Models	B-3	B-4	MT	RGL	QA-L	QA-G	QPP	BERTscore	C-Rel	CA-Rel	CAQ-Rel	Fluency
Baseline	23.98	18.44	21.79	45.95	**65.60**	72.23	26.90	67.62	85.23	89.59	51.19	-10.97
Meteor	25.56	19.84	22.80	47.23	65.52	72.81	28.84	68.36	86.34	90.70	52.54	-10.96
QA-loss	**25.88**	**20.13**	22.93	**47.51**	65.48	74.99	30.57	68.43	85.28	93.81	54.34	-10.95
QA-geo	24.82	19.23	22.23	46.49	65.52	**75.04**	28.06	67.81	86.51	90.39	51.46	-10.96
QPP	25.76	20.08	**22.99**	47.47	65.53	73.87	**31.68**	**68.53**	**88.93**	91.37	54.71	-11.02
BERTscore	24.84	19.27	22.25	46.89	65.59	72.33	28.00	68.18	85.29	90.38	52.97	-10.98
C-Rel	24.71	19.11	22.03	46.65	65.47	71.88	27.33	67.85	87.05	89.83	53.00	-11.00
CA-Rel	24.00	18.51	21.81	46.29	65.34	73.16	28.54	67.26	84.29	**94.55**	**56.22**	-11.04
CAQ-Rel	24.24	18.68	21.92	46.01	65.41	72.56	27.29	67.61	84.53	90.01	52.13	-10.97
fluency	24.19	18.67	21.77	46.11	65.59	71.86	26.47	67.59	84.95	89.61	51.75	**-10.95**

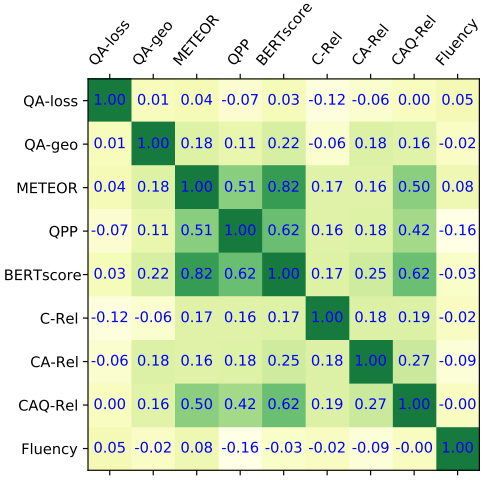

Figure 2: Pearson correlation coefficient matrix of the rewards.

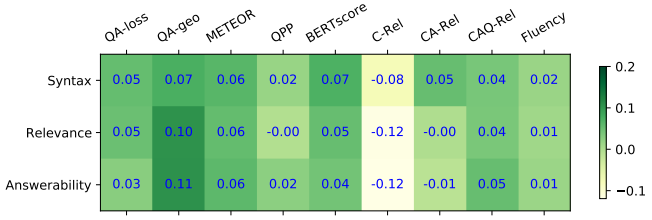

Figure 3: Pearson correlation coefficient matrix between reward scores and human ratings.

To this end, we randomly sampled 100 testing documents, and three computer science students rated questions generated by 9 different models in a blind setup (i.e. they did not receive information on which question was generated by which model): the basic QG model, and the models trained with our different reward functions.

Table 5: Human evaluation results. *GT* means *Ground Truth*. Shown in bold is the best measure for each of the three evaluation dimensions. The ground truth row is not included here.

Reward	Syntax (1/2/3)	Relevance (1/2/3)	Answerability (0/1)
GT	2.86	2.84	0.93
Baseline	2.49	2.32	0.67
Meteor	**2.55**	2.35	0.67
Fluency	2.47	2.18	0.63
QA-loss	2.50	**2.39**	**0.72**
QA-geo	2.41	2.20	0.66
BERTscore	2.50	2.24	0.69
QPP	2.36	2.31	0.68
C-Rel	2.39	2.19	0.61
CA-Rel	2.30	2.22	0.63
CAQ-Rel	2.40	2.22	0.63

We also included the ground-truth question in the labeling process as a control setting as we expect these questions to receive the highest scores in a human evaluation. In order to rate each sample, we provided the context, the ground truth answer span and all the questions for each sample on one screen.

The rating was conducted along three criteria: the *Syntax* (on a scale of 1-3), the *Relevance* (on a scale of 1-3), and the *Answerability* (a boolean value). For syntax, 1 means major syntax issues; 2 means a small mistake (e.g., lacking an article or pronoun); 3 is correct. In the relevance category, 1 means the question is not relevant to the context and the answer; 2 means it is partially relevant (e.g., a question may be more general than what the answer is about); 3 means the question is relevant and relevant to the given answer. In terms of answerability, it needs to be rated whether the question can be answered with the context information and the provided answer. To provide the reader with an intuition, we report three examples

of generated questions with syntax/relevance/answerability issues in Table 1, **Example 2**. As all raters rated the same 100 samples, we considered their average rating for each dimension.

We report the human evaluation results in Table 5. In addition, in Figure 3 we present the correlation between the reward scores and the human ratings.

We make the following observations.

(1) The baseline (i.e. no reward, just the likelihood loss) outperforms all relevance based rewards. Although optimizing on relevance based rewards (C-Rel, CA-Reland CAQ-Rel) leads to improvement of the automatic rewards, it reduces the human rating with respect to syntax, relevance and answerability.
(2) We also add Meteor for the comparison of the performance of n-gram based rewards. We find that optimizing on the Meteor rewards improves all of the three rating criteria. It achieves the best syntax score. As we show in Figure 2 for the automatic evaluation, Meteor is strongly correlated with BERTscore, QPP and CAQ-Rel rewards. This implies that Meteor can capture the lexical and semantics similarity in a way, and can be used as a computation-efficient reward for QG.
(3) The BERT-QA based answerability reward BERT-QA-loss outperforms all other rewards in terms of both Relevance and Answerability. This shows that the BERT-QA-loss metric is a good indicator that reflects the questions' relevance and answerability. This also shows that the QG task is different to common text generation tasks like machine translation or summary generation; here, answerability is a critical criteria for question quality evaluation. Although BERT-QA-geo does not perform as well as BERT-QA-loss, as shown in Figure 3, BERT-QA-geo is most correlated to the human judgment on answerability, relevance and syntax. As the BERT-QA-geo reward is a heuristic indicator for a question's answerability and it does not require answer information, this correlation between the BERT-QA-geo reward and the human judgments implies that it is possible to develop an indicator based on BERT-QA-geo for unsupervised or semi-supervised QA/QG training.
(4) In general, the correlation between the human evaluation dimensions (syntax, relevance, answerability) and the reward scores is rather low: the linear correlation coefficient reaches 0.11 (between answerability and BERT-QA-geo) at best. One reason is of course the very different scoring system (binary or three levels for the human evaluation dimensions). At the same time though, this lack of a high correlation between human ratings and reward scores shows that the reward functions we use are vastly different from the human rating dimensions.

5 CONCLUSIONS

In this work we consider the task of question generation. We systematically categorized past reinforcement learning reward functions proposed for question generation. We implemented all these rewards—as well as three we proposed ourselves—in a common framework to enable a fair evaluation. We performed both an automatic evaluation (with established metrics commonly employed for QG evaluation) as well as a human evaluation, where human raters evaluated the generated questions along the dimensions of syntax, relevance and answerability.

We found that it is indeed effective to apply reinforcement learning on QG model training in terms of the automatic metrics. Overall, the BERT-QA-loss and QPP rewards had the best effectiveness. Our human evaluation showed that BERT-QA-loss achieves also the highest relevance and answerability scores, while using Meteor as reward achieves the highest syntax rating.

In future work we plan to expand this analysis to other datasets, as we now restricted ourselves to SQuAD. As one of our application goals is the setup of an automatic question generator to aid web search users in their learning *whilst* searching the web, we also aim to include a component in the question generator that allows us to change the difficulty of the generated question. And finally, we aim to evaluate how such an "infinite quiz engine" will be received by web search users—does it actually improve human learning?

REFERENCES

[1] Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2019. Reinforcement learning based graph-to-sequence model for natural question generation. *arXiv preprint arXiv:1908.04942* (2019).
[2] Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. 2019. Sentence Mover's Similarity: Automatic Evaluation for Multi-Sentence Texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 2748–2760. https://doi.org/10.18653/v1/P19-1264
[3] Kevyn Collins-Thompson, Preben Hansen, and Claudia Hauff. 2017. Search as learning (dagstuhl seminar 17092). In *Dagstuhl reports*, Vol. 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
[4] Guy Danon and Mark Last. 2017. A syntactic approach to domain-specific automatic question generation. *arXiv preprint arXiv:1712.09827* (2017).
[5] Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*. 376–380.
[6] 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* (2018).
[7] Bhuwan Dhingra, Danish Pruthi, and Dheeraj Rajagopal. 2018. Simple and effective semi-supervised question answering. *arXiv preprint arXiv:1804.00720* (2018).
[8] Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. *arXiv preprint arXiv:1708.06022* (2017).
[9] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*. 13042–13054.
[10] Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106* (2017).
[11] Alexander R Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. Template-based question generation from retrieved sentences for improved unsupervised question answering. *arXiv preprint arXiv:2004.11892* (2020).
[12] Zhihao Fan, Zhongyu Wei, Siyuan Wang, Yang Liu, and Xuanjing Huang. 2018. A Reinforcement Learning Framework for Natural Question Generation using Bi-discriminators. *Coling* (2018), 1763–1774. https://aclanthology.info/papers/C18-1150/c18-1150
[13] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 758–764.
[14] Yifan Gao, Piji Li, Irwin King, and Michael R Lyu. 2019. Interconnected question generation with coreference alignment and conversation flow modeling. *arXiv preprint arXiv:1906.06893* (2019).
[15] Jing Gu, Mostafa Mirshekari, Zhou Yu, and Aaron Sisto. 2021. ChainCQG: Flow-Aware Conversational Question Generation. *arXiv preprint arXiv:2102.02864* (2021).
[16] Michael Heilman. 2011. Automatic factual question generation from text. *Language Technologies Institute School of Computer Science Carnegie Mellon University*

195 (2011).
[17] Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 609–617.
[18] Tom Hosking and Sebastian Riedel. 2019. Evaluating Rewards for Question Generation Models. (2019), 2278–2283. https://doi.org/10.18653/v1/n19-1237 arXiv:1902.11049
[19] Tom Hosking and Sebastian Riedel. 2019. Evaluating rewards for question generation models. *arXiv preprint arXiv:1902.11049* (2019).
[20] Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. A framework for automatic question generation from text using deep reinforcement learning. *arXiv preprint arXiv:1808.04961* (2018).
[21] Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education* 30, 1 (2020), 121–204.
[22] Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 889–898.
[23] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.
[24] David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*. 105–114.
[25] Yanxiang Ling, Fei Cai, Honghui Chen, and Maarten de Rijke. 2020. Leveraging Context for Neural Question Generation in Open-domain Dialogue Systems. In *Proceedings of The Web Conference 2020*. 2486–2492.
[26] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. 2020. Asking questions the human way: Scalable question-answer generation from text corpus. In *Proceedings of The Web Conference 2020*. 2032–2043.
[27] Bang Liu, Mingjun Zhao, Di Niu, Kunfeng Lai, Yancheng He, Haojie Wei, and Yu Xu. 2019. Learning to Generate Questions by LearningWhat not to Generate. In *The World Wide Web Conference*. 1106–1118.
[28] Lin Ma and Yuchun Ma. 2019. Automatic Question Generation based on MOOC Video Subtitles and Knowledge Graph. In *Proceedings of the 2019 7th International Conference on Information and Education Technology*. 49–53.
[29] Xiyao Ma, Qile Zhu, Yanlin Zhou, Xiaolin Li, and Dapeng Wu. 2019. Improving Question Generation with Sentence-level Semantic Matching and Answer Position Inferring. *arXiv preprint arXiv:1912.00879* (2019).
[30] Karen Mazidi and Rodney Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 321–326.
[31] Ruslan Mitkov et al. 2003. Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing*. 17–22.
[32] Jack Mostow and Wei Chen. 2009. Generating Instruction Automatically for the Reading Strategy of Self-Questioning.. In *AIED*. 465–472.
[33] Preksha Nema and Mitesh M Khapra. 2018. Towards a better metric for evaluating question generation systems. *arXiv preprint arXiv:1808.10192* (2018).
[34] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.
[35] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).
[36] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).
[37] Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. *arXiv preprint arXiv:1707.06875* (2017).
[38] 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 of the Association for Computational Linguistics*. 311–318.
[39] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[40] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. *arXiv preprint arXiv:1806.03822* (2018).
[41] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).
[42] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* (2015).
[43] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7008–7024.
[44] Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058* (2017).
[45] Rohail Syed, Kevyn Collins-Thompson, Paul N Bennett, Mengqiu Teng, Shane Williams, Dr Wendy W Tay, and Shamsi Iqbal. 2020. Improving Learning Outcomes with Gaze Tracking and Automatic Question Generation. In *Proceedings of The Web Conference 2020*. 1693–1703.
[46] 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* (2017).
[47] Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yibo Sun, Shujie Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to collaborate for question answering and asking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 1564–1574.
[48] Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018. Learning to ask questions in open-domain conversational systems with typed decoders. *arXiv preprint arXiv:1805.04843* (2018).
[49] Zichao Wang, Andrew S Lan, Weili Nie, Andrew E Waters, Phillip J Grimaldi, and Richard G Baraniuk. 2018. QG-net: a data-driven question generation model for educational content. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. 1–10.
[50] Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, and Yansong Feng. 2020. Exploring Question-Specific Rewards for Generating Deep Questions. *arXiv preprint arXiv:2011.01102* (2020).
[51] Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. *arXiv preprint arXiv:1805.11749* (2018).
[52] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600* (2018).
[53] Xuchen Yao, Emma Tosch, Grace Chen, Elnaz Nouri, Ron Artstein, Anton Leuski, Kenji Sagae, and David Traum. 2012. Creating conversational characters using question generation tools. *Dialogue & Discourse* 3, 2 (2012), 125–146.
[54] Go Yasui, Yoshimasa Tsuruoka, and Masaaki Nagata. 2019. Using semantic similarity as reward for reinforcement learning in sentence generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. 400–406.
[55] Qian Yu, Lidong Bing, Qiong Zhang, Wai Lam, and Luo Si. 2019. based Question Generation with Adaptive Instance Transfer and Augmentation. *arXiv preprint arXiv:1911.01556* (2019).
[56] Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012* (2017).
[57] Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of The Web Conference 2020*. 418–428.
[58] Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. *arXiv preprint arXiv:1909.06356* (2019).
[59] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675* (2019).
[60] 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*. 3901–3910.