A Framework for Automatic Question Generation from Text using Deep Reinforcement Learning

Vishwajeet Kumar[1,2,3] , **Ganesh Ramakrishnan**[2] , **Yuan-Fang Li**[3]
[1]IITB-Monash Research Academy
[2]IIT Bombay
[3]Monash University
{vishwajeet,ganesh}@cse.iitb.ac.in, yuanfang.li@monash.edu

Abstract

Asking intelligent and relevant questions is an important capability of conversational systems such as chatbots. Neural network-based approaches represent the state-of-the-art in automatic question generation (QG). In this work, we attempt to strengthen them significantly by adopting a holistic and novel generator-evaluator framework that directly optimizes objectives that reward semantics and structure. In this paper, we present a novel deep reinforcement learning based framework for automatic question generation. The generator of the framework is a sequence-to-sequence model, whereas the evaluator model of the framework evaluates and assigns a reward to each predicted question. The overall model is trained by learning the parameters of the generator network which maximizes the reward.Our framework allows us to directly optimize any task-specific score including evaluation measures such as BLEU, GLEU, ROUGE-L,etc., suitable for sequence to sequence tasks such as QG. Our evaluation shows that our approach significantly outperforms state-of-the-art systems on the widely-used SQuAD benchmark in both automatic and human evaluation.

1 Introduction

Recent years have seen rapid development in conversational systems, as represented by widely-used personal assistants such as Siri, Cortana, and Alexa, as well as a myriad of online chatbots. Asking intelligent and relevant questions is a very important yet challenging tasks for such systems. Question generation (QG) is the task of generating syntactically correct, semantically sound and relevant questions from various input formats such as text, a structured database or a knowledge base [Mannem *et al.*, 2010]. Recent neural network based techniques [Du *et al.*, 2017; Song *et al.*, 2018; Kumar *et al.*, 2018; Zhao *et al.*, 2018] have achieved remarkable success on QG. These methods typically falls under the sequence-to-sequence (Seq2Seq) setup, employing an RNN-based architecture and additional features such as answering encoding, copy and coverage mechanisms.

These state-of-the-art models are not without their disadvantages. They usually are trained to minimise the cross-entropy loss, which ignores the important sequence information. Moreover, as the training set and the test set may not have the same word distribution, the use of cross-entropy loss may make the training process brittle. In this paper, we first present a framework in which a *generator* mechanism that is employed for generating a question-answer pair invokes or pulls the *evaluator* mechanism that is employed for evaluating the generated pair. Our clearly delineated *generator-evaluator* framework lets us (a) easily incorporate several best practices from the above referred previous models in the *generator* while (b) also letting us employ in the *evaluator*, other complex non-decomposable rewards that are consistent with performance measures (such as BLEU and ROUGE) on test data. We also propose some novel reward functions that *evaluate* the syntax of the question and semantics of the question-answer pair in its entirety. More specifically, since the generated question is in anticipation of some specific answer, we find it most natural to incorporate candidate answer generation (using Pointer Networks) alongside QG right in our generator module, so that the evaluator can optionally take into cognizance the conformity of the generated answer to the ground-truth answer, along with text conformity. Likewise, we also incorporate copy and coverage mechanisms for QG into the generator module so that they can be specifically trained by leveraging a suite of holistically designed and structure-sensitive reward functions in the evaluator module.

The Generator

In Table 1, in rows 1 through 4, we illustrate through examples, the incremental benefits of introducing answer prediction and the copy and coverage mechanisms [See *et al.*, 2017] in the generator. The evaluator associated with the corresponding three generator models employs the conventional and simplistic cross-entropy loss. The motivation for answer prediction in the generator module is obvious and will be further discussed in Section 2.1. In row 3 we illustrate the influence of our copy mechanism, where a rare phrase 'new amsterdam' has been rightly picked up in association with the name of the city. We however note that in row 3, the word 'new' has been erroneously repeated twice, since an encoder-decoder based model could generate questions with meaningless repetitions.We introduce a mechanism for discouraging such repetitions in our generator by quantitatively emphasizing the *coverage* of sentence words while decoding. Row 4 shows the improved and relevant question generated by our model trained by incorporating both the copy and coverage mechanisms.

Evaluator

In row 5 of Table 1, we observe the high-quality question that is generated when the simplistic cross-entropy loss in the evaluator

Text: "new york city traces its roots to its 1624 founding as a trading post by colonists of the dutch republic and was named new amsterdam in 1626 ."		
Row	**Model**	**Question generated**
1	Seq2Seq model optimized on vanilla (cross entropy) loss without answer prediction	in what 1624 did new york city traces its roots ?
2	Seq2Seq model optimized on vanilla (cross entropy) loss with answer prediction	what year was new york named ?
3	Copy aware Seq2Seq model	what year was new new amsterdam named ?
4	Coverage and copy aware Seq2Seq model	in what year was new amsterdam named ?
5	Seq2Seq model optimized on BLEU (using RL)	what year was new york founded ?

Table 1: Sample text and questions generated using variants of our model.

Text: "even with the five largest cities in sichuan suffering only minor damage from the quake , some estimates of the economic loss run higher than us $ 75 billion , making the earthquake one of the costliest natural disasters in chinese history ." **Expected answer:** five		
Row	**Model**	**Question generated**
1	GE_{BLEU}	how much did it making for the earthquake of the economic ?
2	$GE_{BLEU+QSS+ANSS}$	how many largest cities in sichuan experience only minor damage from the quake ?
3	GE_{DAS}	how many cities were in sichuan ?
4	$GE_{DAS+QSS+ANSS}$	how many largest cities in sichuan suffering only minor damage from the quake ?
4	GE_{ROUGE}	how much did the economic loss run in sichuan ?
5	$GE_{ROUGE+QSS+ANSS}$	what is the largest cities in sichuan ?

Table 2: Sample text and questions generated using different reward functions, with and without our new QG-specific rewards QSS+ANSS.

is replaced with the more complex and non-decomposable (across words) BLEU reward that accounts for proximity of 'founded' to 'new york'.

In Table 2, we further illustrate the effect of employing other reward functions (described in Section 2.2) in the evaluator. As can be seen, the model that incorporates QG-specific reward functions (QSS and ANSS) generates a significantly better question when compared to the question generated without these rewards.

Limitations of simple decomposable losses: A Seq2Seq model trained using a vanilla cross-entropy loss function (decomposable over words in the question) generates the question *"what year was new york named ?"* (row 1 in Table 1), which is not addressed in the sentence. The passage talks only about the founding of the city and its naming two years later. The inaccuracy of the question is possibly caused by the use of a loss that is agnostic to sequence information. In other words, given its decomposable nature, the cross-entropy loss on the ground-truth question or any of its (syntactically invalid) anagrams will be the same. Moreover, use of the cross-entropy loss in the sequence prediction model could make the process brittle, since the model trained on a specific distribution over words is used on a test dataset with a possibly different distribution to predict the next word given the current predicted word. This creates exposure bias [Ranzato *et al.*, 2015] during training, since the model is only exposed to the data distribution and not the model distribution. Thus, performance suffers due to inadequately evaluating the *structure* of the generated question against the ground-truth question.

The standard metrics for evaluating the performance of question generation models such as BLEU [Papineni *et al.*, 2002], GLEU, and ROUGE-L [Lin, 2004] are based on degree of n-gram overlaps between a generated question and the ground-truth question. It would be desirable to be able to directly optimize these *task-specific metrics*. However, these n-gram based metrics do not decompose over individual words and are therefore hard to optimize. We explicitly employ an evaluator that rewards each generated question based on its conformance to one (or more than one using decomposable attention) questions in the ground-truth set using these possibly non-decomposable reward functions. We find such learning to be a natural instance of reinforcement learning (RL) [Sutton and Barto, 1998] that allows us to use policy gradient to directly optimize task-specific rewards (such as BLEU, GLEU and ROUGE-L), which are otherwise non-differentiable and hard to optimize. In Table 2 we illustrate questions generated using different reward functions. It can be observed that questions generated using combination of standard reward functions with reward functions specific to QG quality (QSS+ANSS) exhibit higher quality.

Contributions We summarize our main contributions as follows:

- A comprehensive, end-to-end **generator-evaluator framework** naturally suited for automated question generation. Whereas earlier approaches employ some mechanism for generating the question, intertwined with an evaluation mechanism, we show that these approaches can benefit from a much clearer separation of the generator of the question from its evaluator.
- A *generator* founded on the **semantics** and **structure** of the question by (a) identifying target/pivotal answers (Pointer Network), (b) recognizing contextually important keywords in the answer (copy mechanism), and (c) avoiding redundancy (repeated words) in the question (coverage mechanism).
- An *evaluator* that (a) directly optimizes for conformity to the **structure** of ground-truth sequences (BLEU, GLEU, etc.), and (b) matches against appropriate questions from a set of ground-truth questions (Decomposable Attention).
- Novel reward functions that ensure that the generated question is relevant to the text and conforms to the encoded answer.

When evaluated on the benchmark SQuAD dataset [Rajpurkar *et al.*, 2016], our system considerably outperforms state-of-the-art question generation models [Du *et al.*, 2017; Kumar *et al.*, 2018; Song *et al.*, 2018] in automatic and human evaluation.

2 Our Approach

Our framework for question generation consists of a generator and an evaluator. From the reinforcement learning (RL) point of view, the generator is the *agent* and the generation of the next word is an *action*. The probability of decoding a word $P_\theta(word)$ gives a stochastic *policy*. On every token that is output, an evaluator assigns a reward for the output sequence predicted so far using the

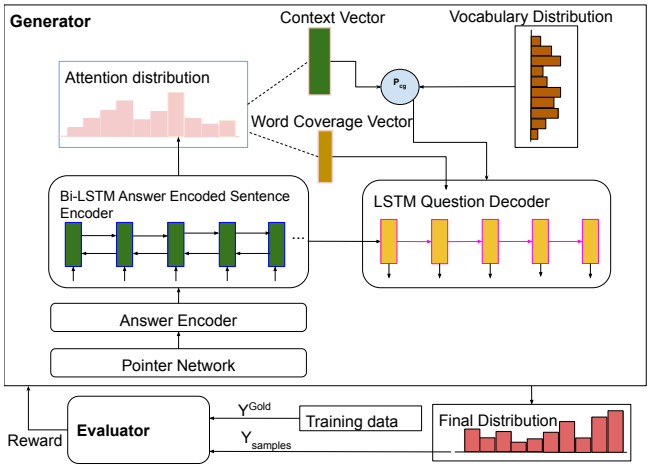

Figure 1: Our generator-evaluator framework for question generation. p_{cg} is the probability which determines whether to copy a word from source text or sample it from vocabulary distribution.

current policy of the generator. Based on the reward assigned by the evaluator, the generator updates and improves its current policy. Let us denote the reward (*return*) at time step t by r_t. The cumulative reward, computed at the end of the generated sequence is represented by $R = \sum_{t=0}^{T} r_t$. The goal of our framework is to determine a generator (policy) that maximizes the expected return:

$$Loss_{RL}(\theta) = -\mathrm{E}_{P_\theta(Y_{0:T}|\mathbf{X})} \sum_{t=0}^{T} r_t(Y_t; \mathbf{X}, Y_{0:t-1}) \qquad (1)$$

where X is the current input and $Y_{0:t-1}$ is the predicted sequence until time $t-1$ and θ is the trainable model parameter. This supervised learning framework allows us to directly optimize task-specific evaluation metrics (r_t) such as BLEU.

The generator is a sequence-to-sequence model, augmented with (i) an encoding for the potentially best pivotal answer, (ii) the copy mechanism [Gu *et al.*, 2016] to help generate contextually important words, and (iii) the coverage mechanism [Tu *et al.*, 2016] to discourage word repetitions. The evaluator provides rewards to fine-tune the generator. The reward function can be chosen to be a combination of one or more metrics. The high-level architecture of our question generation framework is presented in Figure 1.

2.1 Generator

Similar to AutoQG [Kumar *et al.*, 2018], we employ attention and boundary pointer network to identify pivotal answer spans (most important answer spans in the text to ask question about) in the input sentence. The generator then takes as input the sequence of words in the sentence, each augmented with encoding of most probable pivotal answer, along with a set of linguistic features such as POS tag, NER tag, *etc.* At each step, the generator outputs a word with the highest probability, to eventually produce a word sequence. Additionally, as we will see, the generator employs copy and coverage mechanisms.

Sentence Encoder: Each word in the input text is fed sequentially into the encoder along with its linguistic features as well as with the encoded pivotal answer (identified by the boundary pointer network). Our encoder is a two-layer bidirectional LSTM network, consisting of $\overrightarrow{h_t} = \overrightarrow{LSTM_2}(x_t, \overrightarrow{h_{t-1}})$ and $\overleftarrow{h_t} = \overleftarrow{LSTM_2}(x_t, \overleftarrow{h_{t-1}})$, which generates a sequence of hidden states. Here x_t is the given input word at time step t, and $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are the hidden states at time step t for the forward and backward passes respectively.

Question Decoder: Our question decoder is a single-layer LSTM network, initialized with the state $s = [\overrightarrow{h_t}; \overleftarrow{h_t}]$, which is concatenation of hidden state from forward and backward passes.

We also model the attention [Bahdanau *et al.*, 2014] distribution over words in the source text. We calculate the attention (a_i^t) over the i^{th} source word as $a_i^t = softmax(e_i^t)$, where

$$e_i^t = v^t tanh(W_{eh} h_i + W_{sh} s_t + b_{att}) \qquad (2)$$

Here v^t, W_{eh}, W_{sh} and b_{att} are model parameters to be learned, and h_i is the concatenation of forward and backward hidden states of the encoder. We use this attention a_i^t to generate the *context vector* c_t^* as a weighted sum of encoder hidden states: $c_t^* = \sum_i a_i^t h_i$. We further use the c_t^* vector to obtain a probability distribution over the words in the vocabulary as: $P = softmax(W_v[s_t, c_t^*] + b_v)$, where W_v and b_v are model parameters. Thus during decoding, the probability of a word is $P(qword)$. During the training process for each timestamp, the loss is calculated as $L_t = -\log P(qword_t)$. The loss associated with the generated question is:

$$Loss = \frac{1}{T} \sum_{t=0}^{T} L_t = -\frac{1}{T} \sum_{t=0}^{T} \log P(qword_t) \qquad (3)$$

The Copy and Coverage Mechanisms:

The copy mechanism facilitates the copying of important entities and words from the source sentence to the question. We calculate $p_{cg} \in [0,1]$ as the decision of a binary classifier that determines whether to generate (sample) a word from the vocabulary or to copy the word directly from the input text, based on attention distribution a_i^t:

$$p_{cg} = sigmoid(W_{eh}^T c_t^* + W_{sh}^T s_t + W_x x_t + b_{cg}) \qquad (4)$$

Here W_{eh}, W_{sh}, W_x and b_{cg} are trainable model parameters. The final probability of decoding a word is specified by the mixture model:

$$p^*(qword) = p_{cg} \sum_{i:w_i = qword} a_i^t + (1 - p_{cg}) p(qword) \qquad (5)$$

Where $p^*(qword)$ is the final distribution over the union of the vocabulary and the input sentence.

As discussed earlier, Equation (5) addresses the rare words issue, since a word not in vocabulary will have probability $p(qword) = 0$. Therefore, in such cases, our model will replace the <unk> token for out-of-vocabulary words with a word in the input sentence having the highest attention obtained using attention distribution a_i^t.

To discourage meaningless multiple repetitions of words in the question (as illustrated in row 3 of Table 1), we maintain a word coverage vector (wcv) for the words already predicted as

the sum of all the attention distributions ranging over timesteps 0 until $t-1$. Specifically, at time step t, $wcv = \sum_{t'=0}^{t-1} a^{t'}$.

No word is generated before timestep 0, and hence *wcv* will be a zero vector then. After storing the word coverage vector until $t-1$, while attending to the next word, we will need to inform our attention mechanism about words covered until then. Hence, equation (2) is now modified to be:

$$e_i^t = v^t tanh(W_{wcv} wcv_i^t + W_{eh} h_i + W_{sh} s_t + b_{att}) \quad (6)$$

Here W_{wcv} are trainable parameters that inform the attention mechanism about words that have been previously covered while choosing to attend over the next word. Following the incorporation of the copy and coverage mechanism in our generator, the generator's final loss function will be:

$$Loss_{copy+cov} = \frac{1}{T}\sum_{t=0}^{T} \log P^*(w_t) - \lambda_c L_{cov} \quad (7)$$

where λ_c is the coverage hyperparameter and the coverage loss L_{cov} is defined as: $L_{cov} = \sum_i min(a_i^t, wcv_i^t)$

We note that this cross-entropy based loss function still does not include task-specific metrics such as BLEU that were motivated earlier. We employ an evaluator to refine the model pre-trained on this loss function to directly optimize the task specific reward. We also empirically show that the refinement of maximum likelihood models using task-specific rewards such as BLEU improves results considerably. In the next subsection we describe our evaluator.

2.2 Evaluator

The evaluator fine-tunes the parameters of the generator network by optimizing task-specific reward functions through policy gradient. It takes as input the predicted sequence and the gold sequence, evaluates a policy, and returns a reward (a score between 0 and 1) that reflects the quality of the question generated. For question generation, the choice of reward functions include task-specific metrics BLEU, GLEU and ROUGE-L [Du *et al.*, 2017; Kumar *et al.*, 2018], as well as the decomposable attention [Parikh *et al.*, 2016] described below. More importantly, we present two new reward functions that are specifically designed for question generation, QSS and ANSS, for the conformity of questions and answers respectively.

Combining Equation (7) with a reward function R (BLEU, GLEU, ROUGE, DAS, QSS and ANSS), we obtain the overall loss function using the expected reward objective as follows:

$$L_{overall} = \alpha Loss_{copy+cov} + \beta \sum_{t=0}^{T} Loss_{RL}(\theta) \quad (8)$$

where $Loss_{RL}(\theta)$ is reinforcement loss using expected reward (refer to equation 1), $\mathcal{Y}$ is a set of sequences sampled from the final distribution, and α and β are tunable hyperparamters.

Decomposable attention based evaluator

The use of a lexical similarity based reward function such as BLEU or ROUGE does not provide the flexibility to handle multiple possible versions of the ground truth. For example, the questions *"who is the widow of ray croc?"* and *"ray croc was married to whom?"* have almost the same meaning, but due to word order mismatch with the gold question, at most one of them can be rewarded using the BLEU score at the cost of the other(s). Empirically, we find this restriction leading to models that often synthesize questions with poor quality.We therefore, design a novel reward function, a decomposable attention [Parikh *et al.*, 2016] based similarity scorer (DAS). Denoting by $\hat{q}$ a generated question and by q the ground-truth question, we compute a cross attention based similarity using the following steps:

Cross Attention: The generated question $\hat{q}$ and the ground-truth question q are inter-attended as:

$$\hat{q}_i^* = \sum_{j=0}^{L_q} a_{ji} e(q_j),\ a_{ji} = \frac{\exp(e(\hat{q}_i)^T e(q_j))}{\sum_{k=0}^{L_{\hat{q}}} \exp(e(\hat{q}_i)^T e(q_k))},$$
$$q_j^* = \sum_{i=0}^{L_{\hat{q}}} b_{ji} e(\hat{q}_i),\ b_{ji} = \frac{\exp(e(\hat{q}_i)^T e(q_j))}{\sum_{k=0}^{L_q} \exp(e(\hat{q}_k)^T e(q_j))} \quad (9)$$

where $e(.)$ is the word embedding of dimension size d, $\hat{q}^*$ is the cross attention vector for a generated question $\hat{q}$, and q^* is the cross attention vector for a question q in the ground truth.

Comparison: Each n-gram $\hat{q}_i$ in the generated question (through its embedding $e(\hat{q}_i)$) is compared with its associated cross-attention vector $\hat{q}^*$ using a feed forward neural network N_1. Similarly, each n-gram q_j in the ground-truth question (through its embedding $e(q_j)$) is compared with its associated attention vector q^* using another network N_2 having the same architecture as N_1. The motivation for this comparison is that we would like to determine the soft alignment between n-grams in the generated question and the gold question. As an illustration, while comparing the gold question "*why do rockets look white?*" with a generated question "*why are rockets and boosters painted white?*", we find that an n-gram "*rockets and boosters*" is softly aligned to "*rockets*" while "*look*" is softly aligned to "*painted*".

$$\hat{\boldsymbol{q}}_{\boldsymbol{1,i}} = N_1([e(\hat{q}_i), \hat{q}^*]),\ \boldsymbol{q}_{\boldsymbol{2,j}} = N_2([e(q_j), q^*]) \quad (10)$$

where $\hat{\boldsymbol{q}}_{\boldsymbol{1,i}}$ and $\boldsymbol{q}_{\boldsymbol{2,j}}$ are vectors containing comparison scores of aligned phrases in generated question and gold question respectively and N_1 and N_2 are the feed forward neural nets.

Matching Score: The vectors $\hat{\boldsymbol{q}}_{\boldsymbol{1,i}}$ and $\boldsymbol{q}_{\boldsymbol{2,j}}$ are aggregated over each word or phrase in the predicted question and gold question respectively before feeding them to a linear function (L): $DAS = L(\sum_{i=1}^{L_q} \hat{\boldsymbol{q}}_{\boldsymbol{1,i}}, \sum_{j=1}^{L_{\hat{q}}} \boldsymbol{q}_{\boldsymbol{2,j}})$

This matching score between the predicted question and the gold question is the reward returned by the decomposable attention based evaluator.

QG quality specific reward functions

We introduce two new reward functions that specifically designed to evaluate the conformity of the generated question (QSS) and answer (ANSS) against the ground truth.

Question sentence overlap score (QSS): This reward function is specific to QG. We compute the sentence overlap score as the number of common n-grams between predicted question and the source sentence. This reward ensures that generated question is relevant to the given sentence. Thus, if $precision_n(s,q)$ computes the $n-$gram precision match between sentence and question, $QSS = (\prod_{i=1}^{n} precision_i(sentence, question))^{\frac{1}{n}}$

Predicted and encoded answer overlap score (ANSS): In order to ensure that the generated question is about the pivotal

answer/ground truth answer we calculate answer overlap score. Answer overlap score is the number of common n-grams between the encoded answer and the answer predicted (ans_{qa}) for the generated question using the best performing question answering model over SQuAD[1]

$ANSS = (\prod_{i=1}^{n} precision_i(\text{ans}_{qa}, pivotal\_answer))^{\frac{1}{n}}$

3 Experimental Setup

In this section, we present our evaluation framework on the publicly available SQuAD [Rajpurkar *et al.*, 2016] dataset. We first explain various reward functions employed in our experiments. We then describe our baseline and the evaluation methods.

Reward Functions: We experimented with the five reward functions discussed in Section 2.2: (1) BLEU, (2) GLEU, (3) ROUGE-L, (4) DAS, and (5) the QG-specific reward QSS+ANSS. In our experiments we considered BLEU for up to 4-grams. For the GLEU score, we recorded all sub-sequences of up to 4-grams.

Baselines and Evaluation Methods: We reimplemented two state-of-the-art question generation models as baselines for comparison: L2A [Du *et al.*, 2017] and AutoQG [Kumar *et al.*, 2018]. A direct (and fair) comparison with another recent technique, NQG_{LC} [Song *et al.*, 2018], is not feasible, as unlike us, NQG_{LC} requires ground-truth answers, whereas both AutoQG and our model predict pivotal answers. L2A does not consider answers. Moreover, their context (input is sometimes more than one sentence) is different also the train/test split is different from ours. Hence, we only report the original numbers reported in their paper. We also did not perform human evaluation on NQG_{LC} as their source code has not been made available.

We also use an existing implementation of a recent RL-based abstractive summarization technique [Paulus *et al.*, 2018] to train baseline models SUM_{BLEU} (with BLEU as reward function) and $\text{SUM}_{\text{ROUGE}}$ (with ROUGE as reward function). This comparison studies the effectiveness of state-of-the-art abstractive summarization techniques applied to question generation as-is, as the two are conceptually similar tasks.

We report automatic and human evaluation results on eight variants of our model, each of which is equipped with the copy and coverage mechanism, the pointer network, as well as one of the four reward functions: BLEU, GLEU, ROUGE-L, DAS or one of the four rewards in combination with QG quality specific rewards (QSS+ANSS). Hence, our models are named GE_{BLEU}, etc.

For automatic evaluation, we employ BLEU, ROUGE-L and METEOR, which are standard evaluation measures used to evaluate sequence prediction tasks. We use the evaluation scripts released by [Chen *et al.*, 2015] that was originally used to evaluate the image captioning task.

We also performed human evaluation to further analyze the quality of questions generated for their syntactic correctness, semantic correctness and relevance. Syntactic correctness measures the grammatical correctness of a generated question, semantic correctness measures meaningfulness and naturalness of the question, and relevance measures how relevant the question is to the text. We perform human evaluation for each model on a randomly selected subset of 100 sentences. Each of the three judges is presented the 100 sentence-question pairs for each model and asked for a binary response on each quality parameter. The responses from all the judges for each parameter is then averaged for each model.

[1] https://github.com/huggingface/pytorch-pretrained-BERT

4 Results and Discussion

We show and compare results on automatic evaluation in Table 3. Note the numbers in parentheses for L2A [Du *et al.*, 2017], AutoQG [Kumar *et al.*, 2018], and NQG_{LC} [Song *et al.*, 2018] are those reported in their original papers. The slight difference of up to 1.7% in the original and reproduced numbers can be attributed to reimplementation and different versions of various libraries used. As can be seen, all our eight models outperform L2A and AutoQG on all evaluation metrics. Two of our models, GE_{GLEU} and GE_{ROUGE}, also outperform NQG_{LC}. Hence, using evaluation metrics as the reward function during reinforcement based learning improves performance for all metrics. We also observe that $\text{GE}_{\text{ROUGE+QSS+ANSS}}$, the model reinforced with ROUGE-L (that measures the longest common sequence between the ground-truth question and the generated question) as the reward function in combination with QG quality specific rewards(QSS+ANSS), is the best performing model on all metrics, outperforming existing baselines considerably. For example, it improves over AutoQG on BLEU-4 by 29.98%, on METEOR by 13.15%, and on ROUGE-L by 8.67%.

In Table 4 we present human evaluation results for the models evaluated on three quality parameters (a) syntactic correctness, (b) semantic correctness, and (c) relevance.

Consistent with automatic evaluation results shown in Table 3, seven of our eight models outperform the two baselines, with $\text{GE}_{\text{DAS+QSS+ANSS}}$ being the best model on syntactic correctness and semantic correctness quality metrics, outperforming all the other models by a large margin. However, model $\text{GE}_{\text{BLEU+QSS+ANSS}}$ generates highly relevant questions and is the best model on relevance metrics.

It is noteworthy that for each of our models (e.g. GE_{BLEU}), adding QG-specific rewards (e.g. $\text{GE}_{\text{BLEU+QSS+ANSS}}$) significantly improves question quality in human evaluation, even though there is less noticeable improvements in automatic evaluation. This clearly demonstrates the effectivess of our new QG-specific reward functions.

We measure inter-rater agreement using Randolph's free-marginal multirater kappa [Randolph, 2005]. This helps in analyzing level of consistency among observational responses provided by multiple judges. It can be observed that our quality metrics for all our models are rated as *moderate agreement* [Viera *et al.*, 2005].

4.1 Analyzing Choice of Reward Function

BLEU[Papineni *et al.*, 2002] measures precision and ROUGE[Lin, 2004] measures recall, we believe that cross-entropy loss was already accounting for precision to some extent and using it in conjunction with ROUGE (which improves recall) therefore gives best performance.

DAS calculates semantic similarity between generated question and the gound-truth question. As discussed in section 2.2 DAS will give high reward even though the generated question has low BLEU score. Thus, the performance of the model on automatic evaluation metrics does not improve with DAS as the reward

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
L2A [Du *et al.*, 2017]	43.21 (43.09)	24.77 (25.96)	15.93 (17.50)	10.60 (12.28)	16.39 (16.62)	38.98 (39.75)
AutoQG [Kumar *et al.*, 2018]	44.68 (46.32)	26.96 (28.81)	18.18 (19.67)	12.68 (13.85)	17.86 (18.51)	40.59 (41.75)
NQG_{LC} [Song *et al.*, 2018]	-	-	-	- (13.98)	- (18.77)	- (42.72)
SUM_{BLEU} [Paulus *et al.*, 2018]	11.20-	3.50-	1.21-	0.45-	6.68-	15.25-
SUM_{ROUGE} [Paulus *et al.*, 2018]	11.94-	3.95-	1.65-	0.082-	6.61-	16.17-
GE_{BLEU}	46.84	29.38	20.33	14.47	19.08	41.07
$GE_{BLEU+QSS+ANSS}$	46.59	29.68	20.79	15.04	19.32	41.73
GE_{DAS}	44.64	28.25	19.63	14.07	18.12	42.07
$GE_{DAS+QSS+ANSS}$	46.07	29.78	21.43	16.22	19.44	42.84
GE_{GLEU}	45.20	29.22	20.79	15.26	18.98	43.47
$GE_{GLEU+QSS+ANSS}$	47.04	30.03	21.15	15.92	19.05	43.55
GE_{ROUGE}	47.01	30.67	21.95	16.17	19.85	43.90
$GE_{ROUGE+QSS+ANSS}$	**48.13**	**31.15**	**22.01**	**16.48**	**20.21**	**44.11**

Table 3: Experimental results on the test set on automatic evaluation metrics. Best results for each metric (column) are **bolded**. The numbers in parentheses for L2A, AutoQG and NQG_{LC} are those from the best models reported in their respective original papers. The slight difference of up to 1.7% from our reproduced numbers can be attributed to reimplementation and different versions of various libraries used. Models with new QG-specific reward functions (QSS+ANSS) are highlighted in gray for easy comparison.

Model	Syntax		Semantics		Relevance	
	Score	Kappa	Score	Kappa	Score	Kappa
L2A	39.2	0.49	39	0.49	29	0.40
AutoQG	51.5	0.49	48	0.78	48	0.50
GE_{BLEU}	47.5	0.52	49	0.45	41.5	0.44
$GE_{BLEU+QSS+ANSS}$	82	0.63	75.3	0.68	**78.33**	0.46
GE_{DAS}	68	0.40	63	0.33	41	0.40
$GE_{DAS+QSS+ANSS}$	**84**	0.57	**81.3**	0.60	74	0.47
GE_{GLEU}	60.5	0.50	62	0.52	44	0.41
$GE_{GLEU+QSS+ANSS}$	78.3	0.68	74.6	0.71	72	0.40
GE_{ROUGE}	69.5	0.56	68	0.58	53	0.43
$GE_{ROUGE+QSS+ANSS}$	79.3	0.52	72	0.41	67	0.41

Table 4: Human evaluation results (column "Score") as well as inter-rater agreement (column "Kappa") for each model on the test set. The scores are between 0-100, 0 being the worst and 100 being the best. Best results for each metric (column) are **bolded**. The three evaluation criteria are: (1) syntactically correct (Syntax), (2) semantically correct (Semantics), and (3) relevant to the text (Relevance). Models with new QG-specific reward functions (QSS+ANSS) are highlighted in gray for easy comparison.

function, though the quality of questions certainly improves. Further, ROUGE in conjunction with the cross entropy loss improves on recall as well as precision whereas every other combination overly focuses only on precision.

Error analysis of our best model reveals that most errors can be attributed to intra-sentence dependencies such as co-references, concept dependencies *etc.* In a camera ready version of the paper, we will share link to a detailed report containing extensive experiments that include ablation tests. Also link to the source code will be provided then.

5 Related Work

Neural network-based methods represent the state-of-the-art in automatic question generation (QG) from text. Motivated by neural machine translation, Du et al [2017] proposed a sequence-to-sequence (Seq2Seq) architecture for QG. Kumar et al [2018] proposed to augment each word with linguistic features and encode the most relevant *pivotal answer* to the text while generating questions. Similarly, Song et al [2018] encode ground-truth answers (given in the training data), use the copy mechanism and additionally employ context matching to capture interactions between the answer and its context within the passage. They encode ground truth answer for generating questions which might not be available for test set in contrast we train a Pointer Network based model to predict the pivotal answer to generate question about.

Very recently deep reinforcement learning has been successfully applied to natural language generation tasks such as abstractive summarization [Paulus *et al.*, 2018; Celikyilmaz *et al.*, 2018] and dialogue generation [Li *et al.*, 2016]. In summarization, one generates and paraphrases sentences that capture salient points of the text. On the other hand, generating questions additionally involves determining question type such as what, when, etc., being selective on which keywords to copy from the input into the question, leaving remaining keywords for the answer. This also requires the development of a specific probabilistic generative model. [Yao *et al.*, 2018] proposed generative adversarial network (GAN) framework with modified discriminator to predict question type. Recently Fan et al [2018] proposed a bi-discriminator framework for visual question generation. They formulate the task of visual question generation as a language generation task with some linguistic and content specific attributes.

6 Conclusion

We presented a novel, holistic treatment of question generation (QG) using a generator-evaluator framework. Our generator provisions for explicitly factoring in question syntax and semantics, identifies pivotal answers, recognizes contextually important words and avoids meaningless repetitions. Our evaluator allows us to directly optimize for conformity towards the structure of ground-truth question(s). We propose two novel reward functions account for conformity with respect to ground-truth questions and predicted answers respectively. In conjunction, the evaluator makes use of task-specific scores, including BLEU, GLEU, ROUGE-L, and decomposable attention (DAS) that are naturally suited to QG and other seq2seq problems. Experimental results on automatic evaluation and human evaluation on the standard benchmark dataset show that our framework, especially with the incorporation of the new reward functions, considerably outperforms state-of-the-art systems.

References

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

[Celikyilmaz *et al.*, 2018] Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. Deep communicating agents for abstractive summarization. In *NAACL 2016*, pages 1662–1675. ACL, 2018.

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

[Du *et al.*, 2017] Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. In *ACL*, volume 1, pages 1342–1352, 2017.

[Fan *et al.*, 2018] Zhihao Fan, Zhongyu Wei, Siyuan Wang, Yang Liu, and Xuanjing Huang. A reinforcement learning framework for natural question generation using bi-discriminators. In *27th International Conference on Computational Linguistics (COLING)*, pages 1763–1774, 2018.

[Gu *et al.*, 2016] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, volume 1, pages 1631–1640, 2016.

[Kumar *et al.*, 2018] Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. Automating reading comprehension by generating question and answer pairs. In *22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2018.

[Li *et al.*, 2016] Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *EMNLP 2016*, pages 1192–1202. ACL, 2016.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[Mannem *et al.*, 2010] Prashanth Mannem, Rashmi Prasad, and Aravind Joshi. Question generation from paragraphs at UPenn: QGSTEC system description. In *Third Workshop on Question Generation (QG 2000)*, pages 84–91, 2010.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318. ACL, 2002.

[Parikh *et al.*, 2016] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *EMNLP 2016*, pages 2249–2255, 2016.

[Paulus *et al.*, 2018] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *ICLR*, 2018.

[Rajpurkar *et al.*, 2016] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP 2016*, pages 2383–2392. ACL, November 2016.

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

[Ranzato *et al.*, 2015] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[See *et al.*, 2017] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1073–1083, 2017.

[Song *et al.*, 2018] Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. Leveraging context information for natural question generation. In *NAACL (Short Papers)*, volume 2, pages 569–574, 2018.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[Tu *et al.*, 2016] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. In *ACL 2016*, pages 76–85. The Association for Computer Linguistics, 2016.

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

[Yao *et al.*, 2018] Kaichun Yao, Libo Zhang, Tiejian Luo, Lili Tao, and Yanjun Wu. Teaching machines to ask questions. In *IJCAI*, pages 4546–4552, 2018.

[Zhao *et al.*, 2018] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, 2018.