Let's Ask Again: Refine Network for Automatic Question Generation

Anonymous EMNLP-IJCNLP submission

Abstract

In this work we focus on the task of Automatic Question Generation (AQG) where given a passage and an answer the task is to generate the corresponding question. It is desired that the generated question should be (i) grammatically correct (ii) answerable from the passage and (iii) specific to the given answer. An analysis of existing AQG models shows that they produce questions which do not adhere to one or more of these requirements. In particular, the generated questions look like an incomplete draft of the desired question with a clear scope for refinement. To achieve this we propose a method which tries to mimic the human process of generating questions by first creating an initial draft and then *refining* it. More specifically, we propose **Refine Network (RefNet)** which contains two decoders. The second decoder uses a dual attention network which pays attention to both (i) the original passage and (ii) the question (initial draft) generated by the first decoder. In effect, it refines the question generated by the first decoder thereby making it more correct and complete. We evaluate RefNet on three datasets, *viz.*, SQuAD, HOTPOT-QA and DROP, and show that it outperforms existing state-of-the-art methods by 7-16% on all of these datasets. Lastly, we show that we can improve the quality of the second decoder on specific metrics, such as, fluency or answerability by explicitly rewarding revisions that improve on the corresponding metric during training.

1 Introduction

Automatic Question Generation (AQG) is the task of generating a question given a context and optionally an answer. There has been growing interest in AQG as it could help in data curation for Question Answering models, enhance the user experience in conversational AI, and serve as an indicator of the model's ability to comprehend the given context. Given these, it is expected that the questions are (i) grammatically correct (ii) answerable from the passage and (iii) specific to the answer. The existing approaches understand the passage, answer and the relationship between them and then generate the question in a single pass. However, by careful analysis across the generated questions, we observe that these approaches tend to miss out on one or more of these aspects. For instance, in Table 1, the question generated by the baseline system consisting of a single decoder architecture for the first passage is grammatically correct but is not specific to the answer. Moreover, the second example, the generated question is semantically incorrect and incomplete.

Passage 1: Liberated by Napoleon's army in 1806, Warsaw was made the capital of the newly created Duchy of Warsaw.
Generated Questions *Baseline*: What was the capital of the newly duchy of Warsaw? *RefNet*: Who liberated Warsaw in 1806? *Reward RefNet:* Whose army liberated Warsaw in 1806?
Passage 2: To fix carbon dioxide into sugar molecules in the process of photosynthesis, chloroplasts use an enzyme called rubisco
Generated Questions *Baseline*: What does chloroplasts use? *RefNet*: What does chloroplasts use to fix carbon dioxide into sugar molecules? *Reward RefNet*: What do chloroplasts use to fix carbon dioxide into sugar molecules?

Table 1: Samples of generated questions from Baseline, Refine Network and Reward-RefiNet. The ground truth answers are shown in blue

Although these questions are not accurate, revisiting the passage and answer with specific aim to *refine* the initial draft will help in generating a better question in the second pass. We can draw a comparison between this process and how in real life, humans tend to write any piece of text by writing a rough initial draft and refining it over multiple passes. The objective of these revisions could be to improve the already generated draft on various vectors like the fluency or correctness of the generated draft or to make the draft more elaborate. Taking this as motivation, we propose **Refine Networks (RefNet)**, which focuses on improving the question quality by examining the initially generated question and performs a second pass to generate a *revised* question.

Our RefNet is a seq2seq based model that comprises of two decoders: Preliminary and Refinement Decoder. The Refinement Decoder takes the initial draft of the question as an input along with passage and answer, and generates more elaborate and correct questions by simultaneously attending to both the passage and the initial draft. The proposed dual attention aids RefNet to improve upon the initial draft by revisiting the appropriate parts of the input passage in the light of the initial draft. From Table 1, we can infer that our RefNet model is able to generate better questions in the second pass, while simultaneously fixing the errors in the initial draft. We also show that the quality of the *refined* question can be further improved on certain aspects like fluency and answerability by providing explicit reward to RefNet model. This leads to more answerable (Reward-RefNet example for passage 1 in Table 1) and fluent (Reward-RefNet example for passage 2 in Table 1) questions as compared to vanilla RefNet model.

Our experiments show that the proposed RefNet model beats the existing state of the art models for SQuAD dataset by 12.3% and 3.7% (on BLEU) for AQG given the relevant sentence and passage respectively. We also test our model on other Reading Comprehension based datasets (HOTPOT-QA and DROP) and experiments indicate that RefNet is better than the single-decoder baseline by 7.57% and 15.25% (on BLEU). Our human evaluations further validate these results. We further analyze and explain the impact of including the Refinement Decoder by examining the interaction between both the decoders. Interestingly, we observe that the inclusion of the Refinement Decoder boosts the quality of the questions generated by the initial decoder. Lastly, our human evaluation of the questions generated by Reward-RefNet shows their improvement in terms of fluency and answerability of the revised questions over the RefNet questions.

2 Refine Networks (RefNet) Model

In this section, we discuss various components of our proposed model as shown in Figure 1. For a given passage $\mathbf{P} = \{w_1^p, \ldots, w_m^p\}$ of length m and answer $\mathbf{A} = \{w_1^a, \ldots, w_n^a\}$ of length n, we first obtain *answer-aware* latent representation, $\mathbf{U} = \{\tilde{\mathbf{h}}_1^p, \ldots, \tilde{\mathbf{h}}_m^p\}$, for every word of the passage and an answer representation $\mathbf{h}^a$ (as described in Section 2.1). We then generate an initial draft $\tilde{\mathbf{Q}} = \{\tilde{q}_1, \ldots, \tilde{q}_T\}$ by computing $\tilde{q}_t$ as

$$\tilde{q}_t = \arg\max_{\tilde{q}} \prod_{t=1}^{l} \tilde{p}(\tilde{\mathrm{q}}_t|\tilde{\mathrm{q}}_{t-1}, \ldots, \tilde{\mathrm{q}}_1, \mathbf{U}, \mathbf{h}^a)$$

Here $\tilde{p}(.)$ is a probability distribution modeled using the Preliminary Decoder. We then refine the initial draft $\tilde{\mathbf{Q}}$ using the Refinement Decoder to obtain the refined draft $\mathbf{Q} = \{q_1, \ldots q_T\}$:

$$q_t = \arg\max_{q} \prod_{t=1}^{l} p(\mathrm{q}_t|\mathrm{q}_{t-1}, \ldots, \mathrm{q}_1, \tilde{\mathbf{Q}}, \mathbf{U}, \mathbf{h}^a)$$

We then use explicit rewards to enforce refinement on a desired metric, such as, fluency or answerability through our Reward-RefNet model. In the following sub-sections, we describe the passage encoder, preliminary and refinement decoders and our reward mechanism.

2.1 Passage and Answer Encoder

We use a 3 layered encoder consisting of: (i) Embedding, (ii) Contextual and (iii) Passage-Answer Fusion layers as described below. To capture interaction between passage and answer, we ensure that the passage and answer representations are fused together at every layer.

Embedding Layer: In this layer, we compute a d-dimensional embedding for every word in the passage and the answer. This embedding is obtained by concatenating the word's Glove embedding (Pennington et al., 2014) with its character based embedding as discussed in (Seo et al., 2016). Additionally, for passage words, we also compute a positional embedding based on the relative position of the word *w.r.t.* the answer span as described in (Zhao et al., 2018). For every passage word, this positional embedding is also concatenated to the word and character based embeddings. In the subsequent sections, we will refer to embedding of the i-th passage word w_i^p as $e(w_i^p)$ and the j-th answer word w_j^a as $e(w_j^a)$.

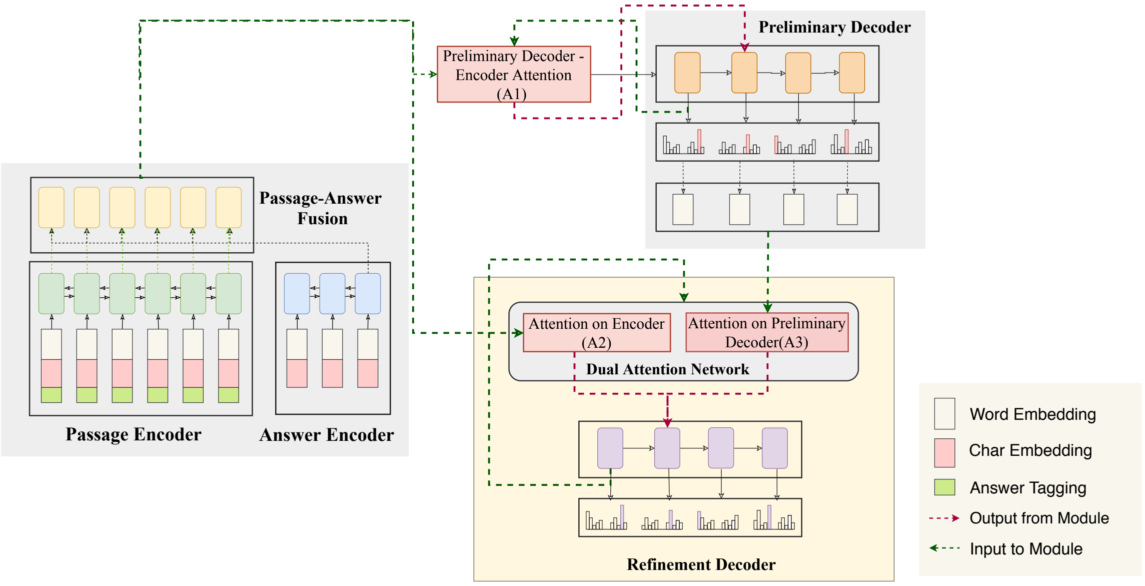

Figure 1: Our RefNet model with Preliminary and Refinement Decoder

Contextual Layer: In this layer, we compute a contextualized representation for every word in the passage by passing the word embeddings (as computed above) through a bidirectional-LSTM (Hochreiter and Schmidhuber, 1997):

$$\overrightarrow{\mathbf{h}_t^p} = \text{LSTM}(e(w_t^p), \overrightarrow{\mathbf{h}^p}_{t-1})$$

where $\overrightarrow{\mathbf{h}_t^p}$ is the hidden state of the forward LSTM at time t. We then concatenate the forward and backward hidden states as $\mathbf{h}_t^p = [\overrightarrow{\mathbf{h}_t^p}; \overleftarrow{\mathbf{h}_t^p}]$.

Similarly, we compute a contextualized representation for the answer words. Note that the n-word answer could also corresponds to a span in the passage. Let $j+1$ and $j+n$ be the start and end indices of the answer span in the passage respectively. We can thus refer to $\{\mathbf{h}_{j+1}^p, \ldots, \mathbf{h}_{j+n}^p\}$ as the representation of the answer words in the context of the passage. We then obtain a refined embedding for the n answer words by passing them through an LSTM as follows:

$$\overrightarrow{\mathbf{h}_t^a} = \text{LSTM}([e(w_t^a), h_{j+t}^p], \overrightarrow{\mathbf{h}^a}_{t-1}) \quad \forall t \in [1, n]$$

The final state $\mathbf{h}^a = [\overrightarrow{\mathbf{h}_n^a}; \overleftarrow{\mathbf{h}_n^a}]$ of this Bi-LSTM is used as the answer representation in the subsequent stages. When the answer is not present in the passage, only $e(w_t^a)$ is passed to the LSTM.

Passage-Answer Fusion Layer: In this layer, we refine the representations of the passage words based on the answer representation as follows:

$$\tilde{\mathbf{h}}_i^p = \tanh\left(\mathbf{W}_u\left[\mathbf{h}_i^p;\ \mathbf{h}^a;\ \mathbf{h}_i^p \odot \mathbf{h}^a\right]\right) \quad \forall i \in [1, m]$$

Here $\mathbf{W}_u \in \mathbb{R}^{l \times 3l}$. l is the hidden size of LSTM. This is similar to how (Seo et al., 2016) capture interactions between passage and the answer for QA. We use $\mathbf{U} = \{\tilde{\mathbf{h}}_1^p, \ldots, \tilde{\mathbf{h}}_m^p\}$ as the answer-aware passage representation which is then used by our decoder(s) to generate the question $\mathbf{Q}$.

2.2 Preliminary and Refinement Decoders

As discussed earlier, RefNet has two decoders, *viz.*, Preliminary Decoder and Refinement Decoder, as described below:

Preliminary Decoder: This decoder generates an initial draft of the question, one word at a time, using an LSTM as follows:

$$\begin{aligned}\tilde{\mathbf{h}}_t^d &= \text{LSTM}([e_w(\tilde{q}_{t-1}); \tilde{\mathbf{c}}_{t-1}; \mathbf{h}^a], \tilde{\mathbf{h}}_{t-1}^d) \\ \tilde{\mathbf{c}}_t &= \sum_{i=1}^{m} \alpha_t^i \tilde{\mathbf{h}}_i^p \end{aligned} \qquad (1)$$

Here $\tilde{\mathbf{h}}_t^d$ is the hidden state at time t, $\mathbf{h}^a$ is the answer representation as computed above, $\tilde{\mathbf{c}}_{t-1}$ is an attention weighted sum of the passage words, α_t^i are parameterized and normalized attention weights (Bahdanau et al., 2014). Let's call this attention network as $\mathbf{A}_1$. $e_w(\tilde{q}_t)$ is the embedding of the word $\tilde{q}_t$. We obtain $\tilde{q}_t$ as:

$$\tilde{q}_t = \arg\max_{\tilde{q}} \left(\text{softmax}(\mathbf{W}_o[\mathbf{W}_c[\tilde{\mathbf{h}}_t^d; \tilde{\mathbf{c}}_t]]\right), \qquad (2)$$

where $\mathbf{W}_c$ is a $\mathbb{R}^{l \times 2l}$ matrix and $\mathbf{W}_o$ is the output matrix which projects the final representation to $\mathbb{R}^V$ where V is the vocabulary size.

Refinement Decoder: Once the preliminary decoder generates the entire question, the refinement decoder uses it to generate an updated version of the question using a **Dual Attention Network**. It first computes an attention weighted sum of the embeddings of the words generated by the first decoder as:

$$g_t = \sum_{i=1}^{T} \beta_i^t e_w(\tilde{q}_i)$$

where β_i^t are parameterized and normalized attention weights computed by attention network $\mathbf{A}_2$. Since the initial draft could be erroneous or incomplete, we obtain additional information from the passage instead of only relying on the output of the first decoder. We do so by computing a context vector $\mathbf{c}_t$ as

$$c_t = \sum_{i=1}^{m} \gamma_i^t \tilde{\mathbf{h}}_i^p$$

where γ_i^t are parameterized and normalized attention weights computed by attention network $\mathbf{A}_3$. The hidden state of the refinement decoder at time t is computed as follows:

$$\mathbf{h}_t^d = \text{LSTM}([e(q_{t-1}); \mathbf{c}_{t-1}; \mathbf{g}_{t-1}; \mathbf{h}^{\mathbf{a}}], \mathbf{h}_{t-1}^d)$$

Finally, q_t is predicted using

$$q_t = \arg\max_q(\text{softmax}(\mathbf{W}_o[\mathbf{W'}_c[\mathbf{h}_t^d; \mathbf{c}_{t-1}; \mathbf{g}_{t-1}]]))$$

where $\mathbf{W}'_c$ is a weight matrix and $\mathbf{W}_o$ is the output matrix which is shared with the Preliminary decoder (Equation 2). Note that when Refinement Decoder is present in RefNet, it generates two variants of the question : initial draft $\tilde{\mathbf{Q}}$ and final draft $\mathbf{Q}$. We compare these two versions of the generated questions in Section 4.

2.3 Reward-RefNet

Next, we address the following question: *Can the refinement decoder be explicitly rewarded for generating a question which is better than that generated by the preliminary decoder on certain desired parameters?*. For example, (Nema and Khapra, 2018) define fluency and answerability as desired qualities in the generated question. They evaluate fluency using BLEU score and answerability using a score which captures whether the question contains the required named entities, verbs and prepositions (and is thus answerable). We use these fluency and answerability scores proposed by (Nema and Khapra, 2018) as reward signals for the refinement decoder. We first compute the reward $r(\tilde{\mathbf{Q}})$ and $r(\mathbf{Q})$ for the question generated by the preliminary and refinement decoder respectively. We then use "REINFORCE with a baseline" algorithm (Williams, 1992) where the Preliminary Decoder's reward $r(\tilde{\mathbf{Q}})$ serves as the baseline. More specifically, given the Preliminary Decoder's generated word sequence $\tilde{\mathbf{Q}} = \{\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_T\}$ and the Refinement Decoder's generated word sequence $\mathbf{Q} = \{q_1, q_2, \dots, q_T\}$ obtained from the distribution $p(\mathrm{q}_t|\mathrm{q}_{t-1}, \dots, \mathrm{q}_1, \tilde{\mathbf{Q}}, \mathbf{U}, \mathbf{h}^a)$, the training loss is defined as follows

$$\begin{aligned} L(\mathbf{Q}) = &(r(\mathbf{Q}) - r(\tilde{\mathbf{Q}}))\cdot \\ &\sum_{t=1}^{T} \log p(q_t|\, q_{t-1}, \dots, q_1, \tilde{\mathbf{Q}}, \mathbf{U}, \mathbf{h}^a) \end{aligned}$$

where $r(\mathbf{Q})$ is the reward obtained by comparing with the reference $\mathbf{Q}^*$. As mentioned, this reward $r(.)$ can be the fluency score or answerability score as defined by (Nema and Khapra, 2018).

2.4 Copy Module

Along with the above-mentioned three modules, we adopt the pointer-network and coverage mechanism from (See et al., 2017). We use it to (i) handle Out-of-Vocabulary words and (ii) avoid repeating phrases in the generated questions.

3 Experimental Details

In this section, we discuss (i) the datasets for which we tested our proposed model, (ii) implementation details and (iii) evaluation metrics used to compare our model with the baseline and existing works.

3.1 Datasets

SQuAD (Rajpurkar et al., 2016): It contains 100K (question, answer) pairs obtained from 536 Wikipedia articles, where the answers are a span in the passage. For SQuAD, AQG has been tried from both sentences and passages. In the former case, the passage is only the sentence which contains the answer span, whereas in the latter case the entire passage is used. We use the same train-validation-test splits as used by (Zhao et al., 2018).
Hotpot QA (Yang et al., 2018) : Hotpot-QA is a multi-document and multi-hop QA dataset. Along with the triplet (P, A, Q), the authors also provide supporting facts that potentially lead to the answer.

Dataset	Model	n-gram						QBLEU4
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR	
SQuAD (Sentence Level)	(Sun et al., 2018)	43.02	28.14	20.51	15.64	-	-	-
	(Zhao et al., 2018)	44.51	29.07	21.06	15.82	44.24	19.67	-
	(Kim et al., 2019)	-	-	-	16.17	-	-	-
	EAD	44.74	29.79	22.00	16.84	44.78	20.60	24.7
	RefNet	**47.27**	**31.88**	**23.65**	**18.16**	**47.14**	**23.40**	**27.4**
SQuAD (Passage Level)	(Zhao et al., 2018)	45.07	29.58	21.60	16.38	44.48	20.25	-
	EAD	44.61	29.37	21.50	16.36	43.95	20.11	24.2
	RefNet	**46.41**	**30.66**	**22.42**	**16.99**	**45.03**	**21.10**	**26.6**
HOTPOT	(Zhao et al., 2018)*	45.29	32.06	24.43	19.29	40.40	19.29	25.7
	EAD	46.00	32.47	24.82	19.68	41.52	23.27	26.2
	RefNet	**45.45**	**33.13**	**26.05**	**21.17**	**43.12**	**25.81**	**28.7**
DROP Dataset	(Zhao et al., 2018)*	39.56	29.19	22.53	18.07	45.01	19.68	31.4
	EAD	39.21	29.10	22.65	18.42	45.07	19.56	31.8
	RefNet	**42.81**	**32.63**	**25.78**	**21.23**	**47.49**	**22.25**	**33.6**

Table 2: Comparsion of RefNet model with existing approaches and EAD model. Here * denotes our implementation of the corresponding work.

The answers here are either yes/no or answer span in P. We concatenate these supporting facts to form the passage. We use 10% of the training data for validation and use the original dev set as test set.

DROP (Dua et al., 2019): The DROP dataset is a reading comprehension benchmark which requires discrete reasoning over passage. It contains 96K questions which require discrete operations such as addition, counting, or sorting to obtain the answer. We use 10% of the original training data for validation and use the original dev set as test set.

3.2 Implementation Details

We use 300 dimensional pre-trained Glove word embeddings, which are fixed during training. For character-level embeddings we initially use a 20 dimensional embedding for the characters which is then projected to 100 dimensions. For answer-tagging embedding size is 3. The hidden size for all the LSTMs is fixed to 512. We use 2-layer, 1-layer and 2-layer stacked BiLSTM for the passage encoder, answer encoder and the decoders (both) respectively. We take the top $30,000$ frequent words as vocabulary size. We use Adam optimizer with a learning rate of 0.0004 and train our models for 10 epochs using cross entropy loss for both the decoders. For the Reward-RefNet model, we fine-tune the pre-trained model with the loss function mentioned in Section 2.3 for 3 epochs. The best model is chosen based on the BLEU (Papineni et al., 2002) score on the validation split. For all the results we use beam search decoding with a beam size of 5.

3.3 Evaluation

We evaluate our models, based on n-gram similarity metrics BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004), and METEOR (Lavie and Denkowski, 2009) using the package released in (Sharma et al., 2017)[1]. We also quantify the answerability of our models using QBLEU-4[2](Nema and Khapra, 2018).

4 Results and Discussions

In this section, we present the results and analysis of our proposed model RefNet. Throughout this section, we refer to our models as follows:

Encode-Attend-Decode (EAD) model is our single decoder model containing the encoder and the Preliminary Decoder described earlier. Note that the performance of this model is comparable to our implementation of the model proposed in (Zhao et al., 2018).

Refine Network (RefNet) model includes the encoder, the Preliminary Decoder and the Refinement Decoder.

We will (i) compare RefNet's performance with existing approaches across all the mentioned datasets (ii) report human evaluations to compare RefNet and EAD (iii) analyze the impact of Refinement Decoder on the performance of Preliminary Decoder and iv) present the performance of

[1] https://github.com/Maluuba/nlg-eval

[2] https://github.com/PrekshaNema25/Answerability-Metric

Reward RefNet with two different reward signal (fluency and Answerability).

4.1 RefNet's performance across datasets

In Table 2, we compare the performance of RefNet with existing single decoder architectures across different datasets. On BLEU-4 metric, RefNet beats the existing state-of-the-art model by 12.30%, 9.74%, 17.48%, and 3.71% respectively on SQuAD (sentence), HOTPOT-QA, DROP and SQuAD (passage) dataset. Also it outperforms EAD by 7.83%, 7.57%, 15.25% and 3.85% respectively on SQuAD (sentence), HOTPOT-QA, DROP and SQuAD (passage). In general, RefNet is consistently better than existing models across all n-gram scores (BLEU, ROUGE-L and METEOR). Along with n-gram scores, we also observe improvements on Q-BLEU4 as well, which as described earlier, gives a measure of both answerability and fluency.

4.2 Human Evaluations

We conducted human evaluations to analyze the quality of the questions produced by EAD and RefNet. We randomly sampled 500 questions generated from the SQuAD (sentence level) dataset and asked the annotators to compare the quality of the generated questions. The annotators were shown a pair of questions, one generated by EAD and one by RefNet from the same sentence, and were asked to decide which one was better in terms of Fluency, Completeness, and Answerability. They were also allowed to skip the question pairs where they could not make a clear choice. Three annotators rated each question and the final label was calculated based on majority voting. We observed that the RefNet model outperforms the EAD model across all three metrics. Over 68.6%, 66.7% and 64.2% of the generated questions from RefNet were respectively more fluent, complete and answerable when compared to the EAD model. However, there are some cases where EAD does better than RefNet. For example, in Table 3, we show that while trying to generate a more elaborate question, RefNet introduces an additional phrase *"in the united"* which is not required. Due to such instances, annotators preferred the EAD model in 30% of the instances.

Passage: Before the freeze ended in 1952, there were only 108 existing television stations in the United States; a few major cities (such as Boston) had only two television stations, ...

Questions

EAD: how many television stations existed in boston ?

RefNet: how many television stations did boston have in the united ?

Table 3: An example where EAD model was better than RefNet. The ground truth answers are shown in blue

Model	**Decoder**	**BLEU-4**	**QBLEU-4**
without $\mathbf{A_3}$	*RefNet*	17.16	25.80
	Initial Draft	17.59	26.00
with $\mathbf{A_3}$	*RefNet*	18.37	27.40
	Initial Draft	17.89	26.00

Table 4: Comparison between Preliminary Decoder and Refinement Decoder in RefNet Model for SQuAD Sentence Level QG

4.3 Impact of Refinement Decoder on Preliminary Decoder

The two decoders impact each other through two paths: (i) indirect path, where they share the encoder and the output projection to the vocabulary V, (ii) direct path, via the dual attention network, where the initial draft of the question is fed to the Refinement Decoder. When RefNet has only indirect path, we can infer from row 1 of Table 4 that the performance of Preliminary Decoder improves when compared to the EAD model (16.84 v/s 17.59 BLEU). This suggests that generating two variants of the question improves the performance of the first decoder pass as well. This is perhaps due to the additional feedback that the shared encoder and output layer get from the Refinement Decoder. Also note that the performance of the Preliminary Decoder is better than Refinement Decoder (row 3). When we add the direct path (attention network) between the two decoders, the performance of the Refinement Decoder improves as compared to the Preliminary Decoder as shown in rows 3 and 4 of the Table 4

Comparison on Answerability: We also evaluate both the initial and refined draft using QBLEU4. As discussed earlier, Q-Metric measures Answerability using four components, *viz.*, Named Entities, Important Words, Function Words, and Question Type. We observe that the increase in Q-Metric for refined questions is because the RefNet model can correct/add the relevant Named Enti-

Sample:	**Sentence:** For instance , the language { xx — x is any binary string } can be solved in linear time on a multi-tape Turing machine , but necessarily requires quadratic time in the model of single-tape Turing machines . **Reference Question:** A multi-tape Turing machine requires what type of time for a solution ?
with $\mathbf{A_3}$	**Refinement Decoder**: in what time can the language be solved on a multi-tape turing machine ? **Preliminary Decoder:** in what time can the language be solved ?
without $\mathbf{A_3}$	**Refinement Decoder:** in what time can the language { xx — x x x x is any binary string ? **Preliminary Decoder:** in what time can the language — x x x x is solved ?

Table 5: Generated samples by Preliminary Decoder and Refinement Decoder in RefNet model.

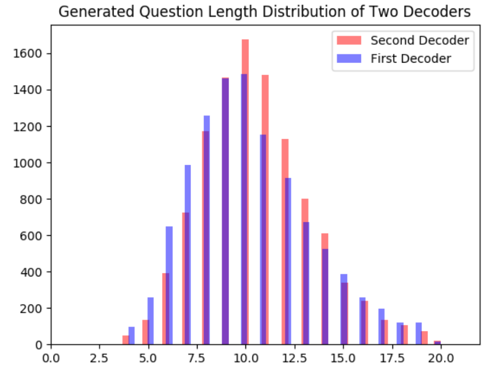

Figure 2: Generated Question Length Distribution for Preliminary Decoder (First Decoder) and Refinement Decoder (Second Decoder)

ties in the question. In particular, we observe that the Named Entity component score in Q-Metric increases from 32.42 for the first draft to 37.81 for the refined draft.

Qualitative Analysis: Figure 2 shows that the RefNet model indeed generates more elaborate questions when compared to the Preliminary Decoder. As shown in Table 5, the quality of the refined question is better than the initial draft of the questions. Here RefNet adds the phrase "multi-tape Turing Machine," which removes any ambiguity in the question.

4.4 Impact of Reward-RefNet

In this section, we analyze the impact of employing different reward signals in Reward-RefNet. As discussed earlier in section 2.3, we use fluency and answerability scores as reward signals. As shown in Table 6, when BLEU-4 (fluency) is used as a reward signal, there is improvement in BLEU-4 scores of Reward-RefNet as compared to RefNet model. We validated these results through human evaluations across 200 samples. Annotators prefer the Reward-RefNet model in 67% of the cases for fluency. Similarly when we use Answerability score as a reward signal, answerability improves for the model and annotators prefer the Reward-RefNet in 70% of the cases for answerability.

Model	**BLEU Reward Signal**		**Answerability Reward Signal**	
	BLEU4	%prefer-ence	Ans.	%prefer ence
RefNet	18.37	32.9	36.9	30
Reward-RefNet	18.52	67.1%	37.5	70%

Table 6: Impact of Reward-RefNet on fluency and answerability. %preference denotes the percentage of times annotators prefer the generated output from the model for fluency in case of BLEU Reward signal and answerability in case of Answerability Reward signal.

Passage: Cost engineers and estimators apply expertise to relate the work and materials involved to a proper valuation
Questions *Generated*: Who apply expertise to relate the work and materials involved to a proper valuation ? *True*: Who applies expertise to relate the work and materials involved to a proper valuation ?

Table 7: An example of question with significant overlap with the passage. The ground truth answers are shown in blue

Case Study: Originality of the Questions

We observe that current state-of-the-art models perform very well in terms of BLEU/QBLEU scores when the actual question has significant overlap with the passage. For example, consider a passage from the SQuAD dataset in table 7, where except the question word *who*, the model sequentially copies everything from the passage and achieves a QBLEU score of 92.4. However, the model performs poorly in situations where the question is novel and does not contain a large sequence of words from the passage itself. In order to quantify this, we first sort the true questions based on its BLEU-2 overlap with the passage in ascending order. We then select the first N data points to compute the QBLEU score between the generated and true questions. The results are shown in red in figure 3. Towards the left, where

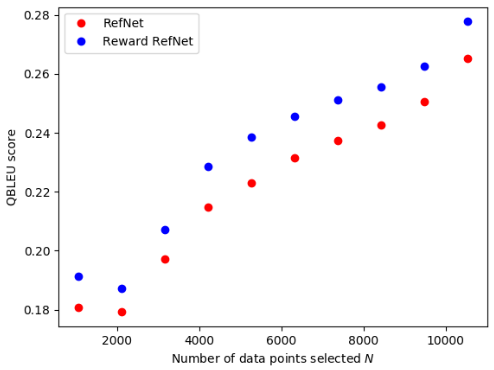

Figure 3: Originality Analysis: Plot of Q-BLEU score vs N - the number points selected

Passage: McLetchie was elected on the Lothian regional list and the Conservatives suffered a net loss of five seats , with leader Annabel Goldie claiming that their support had held firm, nevertheless, she too announced she would step down as leader of the party.

Questions
True: Who announced she would step down as leader of the Conservatives ?
RefNet: who claiming that their support had held firm ?
Reward-RefNet: who was the leader of the conservatives of McLetchie ?

Table 8: An example where Reward-RefNet(Originality) is better than RefNet.

there are true questions with low overlap with the passage, the performance is poor, but it gradually improves as the overlap increases.

The task of generating questions with high *originality* (where the model phrases the question in its own words) is a challenging aspect of AQG since it requires complete understating of the semantics and syntax of the language. In order to improve questions generated on *originality*, we explicitly reward our model for having low n-gram score with the passage as compared to the initial draft. As a result we observe that with Reward-RefNet(Originality), there is an improvement in the performance where the overlap with the passage was less (as shown in blue in figure 3).As shown in Table 8, although both questions are answerable given the passage, the question generated from Reward-RefNet(Originality) is better.

5 Related Work

Early works on Question Generation were essentially rule based systems which did not have a significant learning component (Heilman and Smith, 2010; Mostow and Chen, 2009; Lindberg et al., 2013; Labutov et al., 2015). Current models for AQG are based on the encode-attend-decode paradigm and they either generate questions from the passage alone (Du and Cardie, 2017; Du et al., 2017; Yao et al., 2018) or from the passage and a given answer (in which case the generated question must result in the given answer). Over the past couple of years several variants of the encode-attend-decode model have been proposed. For example, (Zhou et al., 2018) proposed a sequential copying mechanism to explicitly select a sub-span from the passage. Similarly, (Zhao et al., 2018) mainly focuses on efficiently incorporating paragraph level content by using Gated Self Attention and Maxout pointer networks. Some works (Yuan et al., 2017) even use Question Answering as a metric to evaluate the generated questions. There has also been some work on generating questions from images (Jain et al., 2017; Li et al., 2017) and from knowledge bases (Serban et al., 2016; Reddy et al., 2017). The idea of multi pass decoding which is central to our work has been used by (Xia et al., 2017) for machine translation and text summarization albeit with a different objective. Some works have also augmented seq2seq models (Rennie et al., 2017; Paulus et al., 2018; Song et al., 2017) with external reward signals using REINFORCE with baseline algorithm (Williams, 1992). The typical rewards used in these works are BLEU and ROUGE scores. Our REINFORCE loss is different from the previous ones as it uses the first decoder's reward as the baseline instead of reward of the greedy policy.

6 Conclusion and Future Work

In this work, we proposed **Refine Networks (RefNet)** for Question Generation to focus on refining and improving the initial version of the generated question. Our proposed RefNet model consisting of a Preliminary Decoder and a Refinement Decoder with Dual Attention Network outperforms the existing state-of-the-art models on the SQuAD, HOTPOT-QA and DROP datasets. Along with automated evaluations, we also conducted human evaluations to validate our findings. We further showed that using a reward RefNet improves the initial draft on specific aspects like fluency, answerability and originality. As a future work, we would like to extend RefNet to have the ability to decide whether a refinement is needed on the inital draft generated.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. CoRR, abs/1409.0473.

Xinya Du and Claire Cardie. 2017. Identifying where to focus in reading comprehension for neural question generation. In EMNLP, pages 2067–2073. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In ACL (1), pages 1342–1352. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Proc. of NAACL.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In HLT-NAACL, pages 609–617. The Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Comput., 9(8):1735–1780.

Unnat Jain, Ziyu Zhang, and Alexander G. Schwing. 2017. Creativity: Generating diverse questions using variational autoencoders. In CVPR, pages 5415–5424. IEEE Computer Society.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. CoRR, abs/1809.02393.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In ACL (1), pages 889–898. The Association for Computer Linguistics.

Alon Lavie and Michael J. Denkowski. 2009. The meteor metric for automatic evaluation of machine translation. Machine Translation, 23(2-3):105–115.

Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, and Xiaogang Wang. 2017. Visual question generation as dual task of visual question answering. CoRR, abs/1709.07192.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Proc. ACL workshop on Text Summarization Branches Out, page 10.

David Lindberg, Fred Popowich, John C. Nesbit, and Philip H. Winne. 2013. Generating natural language questions to support learning on-line. In ENLG, pages 105–114. The Association for Computer Linguistics.

Jack Mostow and Wei Chen. 2009. Generating instruction automatically for the reading strategy of self-questioning. In AIED, volume 200 of Frontiers in Artificial Intelligence and Applications, pages 465–472. IOS Press.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. CoRR, abs/1808.10192.

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

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. CoRR, abs/1705.04304.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, pages 2383–2392.

Sathish Reddy, Dinesh Raghu, Mitesh M. Khapra, and Sachindra Josh. 2017. Generating natural language question-answer pairs from a knowledge graph using a RNN based question generation model. In EACL (1), pages 376–385. Association for Computational Linguistics.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1179–1195.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In ACL.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. CoRR, abs/1611.01603.

Iulian Vlad Serban, Alberto García-Durán, Çaglar Gülçehre, Sungjin Ahn, Sarath Chandar, Aaron C. Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In ACL (1). The Association for Computer Linguistics.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. CoRR, abs/1706.09799.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. CoRR, abs/1709.01058.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn., 8(3-4):229–256.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 1784–1794. Curran Associates, Inc.

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. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

Kaichun Yao, Libo Zhang, Tiejian Luo, Lili Tao, and Yanjun Wu. 2018. Teaching machines to ask questions. In IJCAI.

Xingdi Yuan, Tong Wang, Çaglar Gülçehre, Alessandro Sordoni, Philip Bachman, Saizheng Zhang, Sandeep Subramanian, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. In Rep4NLP@ACL, pages 15–25. Association for Computational Linguistics.

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

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2018. Sequential copying networks. In AAAI.