Neural Question Generation with Answer Pivot

Bingning Wang, Ting Yao, Qi Zhang, Jingfang Xu, Xiaochuan Wang
Sogou Inc.
Beijing, 100084, China
{wxc,wangbingning,yaoting,qizhang,xujingfang}@sogou-inc.com

Abstract

Neural question generation (NQG) is the task of generating questions from the given context with deep neural networks. Previous answer-aware NQG methods suffer from the problem that the generated answers are focusing on entity and most of the questions are trivial to be answered. The answer-agnostic NQG methods reduce the bias towards named entities and increasing the model's degrees of freedom, but sometimes result in generating unanswerable questions which are not valuable for the subsequent machine reading comprehension system. In this paper, we treat the answers as the hidden pivot for question generation and combine the question generation and answer selection process in a joint model. We achieve the state-of-the-art result on the SQuAD dataset according to automatic metric and human evaluation.

Introduction

Question generation (QG), or learning to ask, is a challenging problem in natural language understanding, it has been an active field of research within the context of machine reading comprehension (MRC). Question generation has many useful applications such as improving the MRC (Yuan et al. 2017; Xiao et al. 2018) by providing more training data, generating educational purposes exercises (Heilman and Smith 2010), and helping dialog systems, such as Alexa and Google Assistant.

Conventional methods for question generation rely heavily on heuristic rules, sometimes the standalone constituent or dependency parsing tool is needed to generate the handcrafted templates (Mostow and Chen 2009; Heilman and Smith 2010; Rus et al. 2010; Hussein, Elmogy, and Guirguis 2014). These rule-based systems are brittle and have low generalizability and scalability. Recent works on question generation are focusing on using deep neural networks with the end to end training, which is also known as neural question generation. NQG is based on sequence to sequence methods, using mechanism borrowed from the neural machine translation, such as copy (Çaglar Gülçehre et al. 2016; Zhou et al. 2017) and attention (Bahdanau, Cho, and Bengio 2014; Yuan et al. 2017; Scialom, Piwowarski, and Staiano 2019). NQG shows great advantage compared with previous rule-based systems both in terms of question fluency and diversity (Duan et al. 2017; Yuan et al. 2017).

Paragraph: In accordance with his father's wishes, Luther enrolled in law school at the same university that year but dropped out almost immediately, believing that law represented uncertainty. Luther sought assurances about life and was drawn to theology and philosophy.

Answer-Aware:	**Predicted Answer:** Luther **Predicted Question:** Who enrolled in law school accord with his father's wishes?
Answer-Agnostic:	**Predicted Question:** Why does Luther sought assurances about life?

Figure 1: The bad case of answer-aware and answer-agnostic NQG. In answer-aware question generation, the generated answer *Luther* is just a named entity that could be trivially inferred by subsequent text, and without much value to be asked. In answer-agnostic case the generated question is fluent but could not be answered by the paragraph.

Briefly, NQG systems are mainly divided into two streams: answer-aware and answer-agnostic. In answer-aware NQG system, the models are given not only the paragraph but also the target answers (Yuan et al. 2017; Sun et al. 2018; Song et al. 2018; Chen, Wu, and Zaki 2019), and the models are learned to interact with the paragraph and the target answer to generate the specific questions. However, in real applications, the answers are not provided so we should first generate the candidate answers and then produce the questions thereof. Dong et al. (2018) found that the generated answers are focused on named entities so the question types are limited to certain types. Furthermore, Golub et al. (2017) showed that sometimes the selected answers are just arbitrary entities, regardless of their importance in the corresponding paragraph,so the generated questions are trivial that benefit little to the MRC systems (Duan et al. 2017).

Conversely, current NQG systems are more and more focusing on answer-agnostic NQG (Subramanian et al. 2018; Kim et al. 2019; Scialom, Piwowarski, and Staiano 2019), which lifts the constraint of knowing the target answers before generating the questions (Du, Shao, and Cardie 2017). The agnostic of the target answer increasing the model's degrees of freedom to generate diverse questions. However, the answer-agnostic NQG systems are suffered from the fact the generated answers may be unanswerable (Sun et al. 2018), an example in Figure 1 demonstrates this problem. Furthermore, the lack of corresponding answers limiting their application in MRC where the answers are requisite.

In this paper, we try to combine the advantage of answer-aware and answer-agnostic NQG in a joint model. We treat the answers as the *hidden pivot* when generating the questions. Concretely, we first generate the hidden answers given the paragraph, and then combined paragraph and the induced pivot answers to produce the questions, the objective is to maximize the likelihood of the questions. In this way, the model is learned such that the better hidden answers pivot could yield better questions. Our model could be seemed as the compromise between answer-aware and answer-agnostic model. If we ignore the hidden answer pivot, then it reduced to answer-agnostic models where the answers are bypassed; if we fed the ground truth answers as the pivot then its behaves like the answer-aware model. Therefore, our model could take the advantages of both worlds.

We conduct throughout experiments on SQuAD (Rajpurkar et al. 2016). The proposed model consistently outperforms the pure answer-aware or answer-agnostic counterparts in terms of the automatic evaluation metric. The human assessment demonstrates that our proposed model could generate both answerable and diverse questions. Furthermore, preliminary experiments show that the induced hidden answers are accord with the real target answers, even if the model was trainined without answer supervision. Finally, the generated data of our model also excels at improving the result of downstream MRC. Codes and analysis of this paper will be public available.

Related Work

Automatic question generation has received increased attention from the research community. Traditional QG systems are most rule-based, which sometimes utilizing off-the-shelf NLP tools to get the syntactic structure, dependency relations and semantic role of the passage (Mostow and Chen 2009; Heilman and Smith 2010; Chali and Hasan 2015). First, the target answers are generated using rules or semantic roles, next, they generate questions using hand-crafted transformation rules or templates. Finally, the generated questions are ranked by features such as key word matching degree or sentence perplexity (Hussein, Elmogy, and Guirguis 2014; Heilman 2011). The main drawbacks of these symbolic systems are that the rules and templates are expensive to manually create, and lack diversity.

Recently, with the development of deep learning and large-scale question answering dataset, motivated by neural machine translation, Du, Shao, and Cardie (2017) proposed a sequence to sequence (seq2seq) architecture combined with attention mechanism, achieving a promising result on MRC dataset SQuAD. Since then, many works have been proposed to extends the preliminary framework with rich features, such as answer position (Sun et al. 2018), named entity tags (Zhou et al. 2017) or templates (Duan et al. 2017), and incorporate copy mechanism to copy words from the context paragraph (Song et al. 2018). However, these methods are all based on the maximum likelihood estimation, which has the notorious problem of exposure bias (Ranzato et al. 2015) and other deficiency during inference (Kumar, Ramakrishnan, and Li 2018; Chen, Wu, and Zaki 2019). Some training objectives other than teacher forcing are introduced, such as BLEU score (Kumar, Ramakrishnan, and Li 2018), generated question perplexity (Yuan et al. 2017) or word embedding similarities (Chen, Wu, and Zaki 2019). However, Hosking and Riedel (2019) found that although those policy gradient methods leads to increases in the metrics such as BLEU, but they are poorly aligned with human judgment and the model simply learns to exploit the weaknesses of the reward source.

While most NQG models are focused on answer-aware setting, recently, answer-agnostic NQG has attracted more and more attention. In the case that only the input passage is given, the system should automatically identify question-worthy parts within the passage and generate questions thereof. Du, Shao, and Cardie (2017) learns a sentence selection task to identify the sentences in the paragraph using a neural network-based sequence tagging models. Subramanian et al. (2018) train a neural keyphrase extractor to predict the keyphrase within the paragraph. Scialom, Piwowarski, and Staiano (2019) argues that the predicted answer may make the generated question biased towards the factoid questions, and they train a Transformer (Vaswani et al. 2017) based answer-agnostic model and obtain a promising result in terms of the human evaluation.

The pros and cons of previous answer-aware and answer-agnostic NQG models motivate us to combine them together: our model are built upon answer-agnostic NQG, but we explictly infer the hidden answers and generated questions based on the induced hidden answer.

Methodology

In this paper, we denote the context paragraph as $C = \{c_1, c_2, ..., c_n\}$, our objective is to predict the target questions $Q = \{q_1, q_2, ..., q_m\}$. The whole architecture is built upon the standard encoder-decoder architecture, with the multi-head attention as the building block, an additional hidden pivot predictor is introduced to get the candidate answer.

Paragraph Encoder

The paragraph encoder encodes the paragraph into dense embedding space. In this paper, based on the current development of NLP (Radford et al. 2019; Devlin et al. 2018), we adopt the self-attention based Transformer (Vaswani et al. 2017) as the building block. The hidden representation for layer l could be represented as:

$$\begin{aligned} \mathbf{q}^l, \mathbf{k}^l, \mathbf{v}^l &= \mathbf{W}_q^l \mathbf{h}^{l-1}, \mathbf{W}_k^l \mathbf{h}^{l-1}, \mathbf{W}_v^l \mathbf{h}^{l-1}, \\ \mathbf{h}^l &= \text{MultiHeadAttention}(\mathbf{q}^l, \mathbf{k}^l, \mathbf{v}^l) \end{aligned} \tag{1}$$

$$\text{MultiHeadAttention}(q, k, v) = softmax(\frac{\mathbf{q}^T \mathbf{k}}{\sqrt{d}})\mathbf{v} \tag{2}$$

$\mathbf{q}^l, \mathbf{k}^l, \mathbf{v}^l$ are query, key and value representations for this layer and $\mathbf{W}^l_{q,k,v}$ are weight matrixes. d is the hidden size. The first layer is the word embedding layer, we use the last layer output $\mathbf{H} \in \mathbb{R}^{n \times d}$ as the paragraph representations.

Pivot Answer Predictor

Compared with prvious anwer-agnostic NQG methods, the most significant difference of our model is that we explicity infer the candidate answer before generating the questions, thus the induced answers act like a *pivot* in our model. The pivot answer predictor predict the hidden answer based on the current paragraph. We predict the binary label of the i_{th} word to denote whether the current tokens is locate within the answer spans:

$$\begin{aligned} \mathbf{z}_i &= \text{MLP}(\mathbf{h}_i), \\ g_i &= \sigma(\mathbf{z}_i^T \mathbf{w}_z), \\ z_i &= \begin{cases} 1, & g_i > 0.5, \\ 0, & g_i \leq 0.5 \end{cases} \end{aligned} \tag{3}$$

where MLP is the multi-layer perceptron, and σ is the sigmoid activation function. **w** is the weight vector to transform the hidden representation into a scalar value. z is a binary indicator to denote the current word is in (1) or out of (0) the answer span. Thus, our model could be fitted to the scenario where the answer spans are continous or discontinues.

After the pivot answer predictor, along with the original paragraph hidden representations **H**, we also has the answer position information which could guide the subsequent decoder to generate specific questions. As the answer indicator z is a binary value, we use an embedding $\mathbf{D} \in \mathbb{R}^{2 \times d}$ matrix to embed this indicator to the representation **Z**, and add it to **H** as the final hidden representation of the encoder:

$$\mathbf{C} = \mathbf{H} + \mathbf{Z} \tag{4}$$

Question Decoder

The question decoder is similar with previous answer-aware NQG model, which takes the paragraph hidden representations and answer indicator as input and generate the target question in an auto-regressive way. The probability of generating the target token in step i is:

$$p(q_i | q_{<i}, \mathbf{C}) \tag{5}$$

where $q_{<i}$ represents the question words at earlier timesteps. We adopt the Transformer as the decoder module. Suppose the inner state of the current timestep in layer $l-1$ is $\mathbf{s}_t^{l-1}$:

$$\begin{aligned} \mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t &= \mathbf{W}_q \mathbf{s}_t^{l-1}, \mathbf{W}_k \mathbf{s}_{\leq t}^{l-1}, \mathbf{W}_v \mathbf{s}_{\leq t}^{l-1}, \\ \bar{\mathbf{s}}_t^l &= \text{MultiHeadAttention}(\mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t) \\ \bar{\mathbf{q}}_t, \bar{\mathbf{k}}_t, \bar{\mathbf{v}}_t &= \overline{\mathbf{W}}_q \bar{\mathbf{s}}_t^l, \overline{\mathbf{W}}_k \mathbf{C}, \overline{\mathbf{W}}_v \mathbf{C}, \\ \mathbf{s}_t^{'l} &= \text{MultiHeadAttention}(\bar{\mathbf{q}}_t, \bar{\mathbf{k}}_t, \bar{\mathbf{v}}_t) \\ \mathbf{s}_t^l &= \mathbf{s}_t^{'l} + \bar{\mathbf{s}}_t^l \end{aligned} \tag{6}$$

The first multi-head attention is the self-attention to gather the decoder information up to current timestep, and the second multi-head attention is to get the attentive representation from the paragraph. Finally their representations are added as the current layer output, and the output probability is:

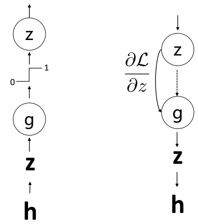


Figure 2: The forward and backward pass in our model. We use the straight through estimator to estimate the gradient of the scalar values z during backward phase.

$$\mathbf{o}_t = softmax(\mathbf{W}_o \mathbf{s}_t^L + \mathbf{b}_o) \tag{7}$$

where $\mathbf{W}_o \in \mathbb{R}^{d \times |V|}$ and $\mathbf{b}_o \in \mathbb{R}^{|V|}$ are the output weight parameters, $|V|$ is the size of the shortlist vocabulary.

Training

The model is initially trained to minimize the negative log-likelihood of the data under the model distribution,

$$\mathcal{L}_{\text{MLE}} = -\mathbb{E}_q \sum_i \log p(q_i | q_{<i}, \mathbf{C}) \tag{8}$$

However, in the answer pivot layer, we make a binary sample from the indicator (Eq. 3), which makes the model discontinuous and the gradient flow is block in standard error back-propagation. To enable the end to end training, a common approach is to use the policy gradient methods adopted from reinforcement learning where the *rewards* are environment-provided scalar values (Williams 1992). However, it suffers from the well-known variance problem of estimating the gradient (Sutton et al. 2000). This problem is even severed in our application because there are n candidate output in a single time, making the variance even higher.

In this paper, inspired by previous works of training deep belief networks and other types of hard non-smooth neural networks (Hinton, Srivastava, and Swersky 2012; Bengio, Léonard, and Courville 2013). We use the straight-through estimator to estimate the gradients for binary latent answer pivot. Suppose the parameter of the encoder is θ, then the gradients are estimated by:

$$\begin{aligned} &\frac{d\mathbb{E}_q \sum_i \log p(q_i | q_{<i}, \mathbf{C})}{\partial \theta} \\ &= \frac{d\mathbb{E}_q \sum_t \log p(q_i | q_{<i}, \mathbf{C})}{dz} \frac{dz}{dg} \frac{dg}{\partial \theta} \\ &\approx \frac{d\mathbb{E}_q \sum_t \log p(q_i | q_{<i}, \mathbf{C})}{dz} \frac{dg}{\partial \theta} \end{aligned} \tag{9}$$

The straight through gradient estimator is shown in Figure 2. We can see that the gradient to the hard neurons z are approximated by its source $g = \sigma(\mathbf{z}^T \mathbf{w}_z)$ which is smooth and derivable. Although it is a biased estimator, it has been shown to be a fast and efficient method to estimate the gradient of the discrete variables, especially for the Bernoulli variable case (Hubara et al. 2016; Shen et al. 2018).

Auxiliary Pivot Prediction

The initial MLE objective in Eq. 8 is powerful to train the encoder-decoder architecture. However, in our preliminary experiment, we found that sometimes our model may ignore the latent pivot variable and generate the questions only based on the context paragraph, which reduce the model to the answer-agnostic scenario. This phenomenon is also referred to posterior collapse problem (Lucas et al. 2019) in variational autoencoder (Kingma and Welling 2013).

In this paper, we propose an auxiliary objective to ameliorate this issue. After the question generation from the decoder, we concatenate the paragraph representations **H** with question representations **S** in Eq. 6, and predict the binary answer pivot labels in the paragraph,

$$\begin{aligned}\overline{\mathbf{H}} &= [\mathbf{H}; \mathbf{S}] \\ \tilde{\mathbf{H}} &= \text{TransformerEncoder}(\overline{\mathbf{H}}) \\ \mathbf{b} &= \sigma(\mathbf{w}_h^T \tilde{\mathbf{H}}_{[0:n]}) \\ \mathcal{L}_{\text{AP}} &= -\sum_i z_i \log b_i + (1 - z_i) \log(1 - b_i)\end{aligned} \tag{10}$$

Where the subscript AP denotes the answer position, and the loss is the binary cross entropy between the predicted answer and pivot answer position. This objective is similar with the answer pivot prediction in Eq. 3 except that we incorporate the question representations. The question representations are obtained from the decoder which also takes the answer pivot information **Z** as input. Therefore, when the model is optimized, the question would take the pivot answer into account to reduce this loss function, making the predicted hidden answer pivot more and more accurate.

The final objective for our model is a linear combination of the maximum likelihood loss in Eq. 8 and Eq. 10, i.e.,

$$\mathcal{L} = \mathcal{L}_{\text{MLE}} + \lambda \mathcal{L}_{\text{AP}} \tag{11}$$

Supervised Training with Golden Answers

The proposed methods only requires the paragraph-question pairs, and the pivot answers are inferred by the model itself. In some NQG applications we also had the ground truth answers. Inspired by the supervised attention (Liu et al. 2016), we can improve the pivot answer predictor with the additional supervision. Concretely, given the ground truth answers labels $a_i \in \{0, 1\}$, which denote whether the i_{th} tokens is in the answers, after the pivot answer prediction layer in Eq. 3, we add the additional supervised objective:

$$\mathcal{L}_{\text{S}} = -\sum_i a_i \log g_i + (1 - a_i) \log(1 - g_i) \tag{12}$$

which would motivate the predicted pivot answers towards the ground truth answers. Adding this objective we get the supervised loss function to optimize:

$$\mathcal{L} = \mathcal{L}_{\text{MLE}} + \lambda_1 \mathcal{L}_{\text{AP}} + \lambda_2 \mathcal{L}_{\text{S}} \tag{13}$$

Experiment

Dataset

In this paper, we conduct the experiments on the SQuAD dataset that has been widely used for NQG evaluation. The SQuAD dataset consists of 23,215 paragraphs from 536 articles in Wikipedia, with nearly 100,000 crowd-sourced question-answer pairs. 87,600 questions are used for training, 10,570 for development, and an unknown number in a hidden test set. Since the test sets are not publicly available, we follow Zhou et al. (2017) to randomly split the dev set into two parts and use them as the development set and test set for NQG. Thus, the total number of training, developing and testing set is 86,635, 8,965 and 8,964 respectively.

Implementation Details

In this paper, we preprocess the text with the sentencepiece (Kudo and Richardson 2018) tokenizer with vocabulary size 30,000. We initialize the word embedding with the skip-gram algorithm[1]. We truncate the paragraph to max sequence size of 256, and question to max sequence length of 30. For the encoder and decoder, we set the number of layers to 4, and the number of head to 6, hidden size is set to 384. Dropout is applied to the output of word embedding layer and the multi-head attention layer with rate 0.2. We use the Adam (Kingma and Ba 2014) optimizer with default hyperparameters to optimize the models. During inference, we adopt a top-5 beam search with length penalty of 0.9.

For the answer pivot weight λ in Eq. 11, we first optimize the MLE loss and then gradually increasing it from 0 to 0.5, which is tuned in the development set. We found this training strategy would reduce the loss variance, therefore, the updates of the parameters are smooth.

Following previous works of NQG (Song, Wang, and Hamza 2017; Zhou et al. 2017; Chen, Wu, and Zaki 2019), we adopt 3 automatic evaluation metrics: **BLEU**, **Meteor** and **Rouge-L**, which measure the n-gram similarities between the generated questions and real questions.

answer-agnostic NQG Result

We first evaluate our model on the pure answer-agnostic setting, where we have no ground truth answers during training and inference period. We adopt two answer-agnostic NQG models for comparison:

- **L2A** (Du, Shao, and Cardie 2017) is the very first NQG models which is a standard standard seq2seq model with attention mechanism.
- **SAQG** (Scialom, Piwowarski, and Staiano 2019) is a self-attention based answer-agnostic models enhanced with copy mechanism. SAQG also replace the named entity with a placeholding tokens.

The result is shown in Table 1. Although our model is not equipped with the sophisticated mechanism such as copy or placeholding, but our vanilla model is comparative with the previous state of the art answer-agnostic models. When incorporating the answer pivot prediction layer, the performance of our model boost a lot and achieves a new state of the art result on the answer-agnostic NQG. It demonstrates the advantage and effectiveness of explicitly incorporating answer information when generating the questions.

[1] https://code.google.com/archive/p/word2vec/

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor	Rouge-L
answer-agnostic						
L2A	43.09	25.96	17.50	12.28	16.62	39.75
SAQG	43.33	26.27	18.32	13.23	-	40.22
Our model w/o AP	43.18± 0.57	25.99 ± 0.36	18.67 ± 0.25	13.09± 0.18	17.24± 0.24	40.98 ± 0.37
Our model	**47.24 ± 0.67**	**28.03± 0.42**	**20.96± 0.29**	**14.78 ± 0.17**	**18.61± 0.28**	**41.87± 0.27**
answer-aware						
NQG++	44.82	26.06	18.28	13.02	16.68	41.22
SynNet	47.98	28.08	20.45	15.27	17.53	41.91
KPG	46.23	27.78	20.95	14.93	17.24	40.48
ASs2s	47.09	28.62	19.04	15.02	17.02	41.89
Our model	**48.26 ± 0.51**	**29.23± 0.30**	**22.37± 0.19**	**16.42 ± 0.13**	**18.95 ± 0.21**	**43.07± 0.19**

Table 1: Experimental results on answer-agnostic answer-aware setting. *Our model w/o AP* refers to our model without the answer pivot layer. In answer-aware setting the ground truth answers are only provided during training. The performances of our models are reported as the mean and standard derivation values run on five different initiation.

answer-aware NQG Results

We also compare our model with the answer-aware NQG models where the ground truth answers are provided. Before reporting the results, we note that the previous NQG models sometimes assume the answers are given in both training and inference period. However, in real inference applications, the ground-truth answers are not provided, so we should generate the answer in advance. This step is also called *answer selection* in previous NQG (Xiao et al. 2018; Subramanian et al. 2018). For fairly comparison, we adopt 4 NQG baselines that have public implementations:

- **NQG++**[2] (Zhou et al. 2017) is a GRU based seq2seq model with copy and attention mechanism, enhanced with answer position features and lexical features. We adopt the same answer selection model with (Xiao et al. 2018).
- **SynNet**[3] (Golub et al. 2017) is an LSTM based seq2seq model. In addition to the question generation, they also adopt an IOB tagging model to predict the answer.
- **KPG**[4] (Subramanian et al. 2018) also takes the answer prediction into account. Before generating the question they first extract the key phrase in the document.
- **ASs2s**[5] (Kim et al. 2019) is also based on LSTM. They mask the answers with a special token to reduce the answer's appearance in the target question. We predict the special mask tokens during inference.

The result is shown in Table 1. Our model also excels at the answer-aware setting. In the previous models they are always fed with the ground-truth answers during training, however, during inference the answers are predicted, the training and inference discrepancy is also referred as exposure bias in machine translation (Ranzato et al. 2015), and the predicted answers are sometimes poor that hurt quality of the question generated on it. On the contrary, in our proposed answer pivot models, the questions are always generated based on the induced answers that bypass the exposure bias problem, and the supervised signal further improves the answer generation quality.

Human Evaluation

Although the automatic evaluation is an efficient criterion to evaluate the quality of the NQG systems, sometimes they are biased toward a specific attribute of the generated question (Hosking and Riedel 2019). So we conduct human qualitative evaluation of the generated outputs. We randomly sample 100 context-question pairs from the test set and ask three volunteers to evaluate the sample quality. We consider three aspects of the generated questions:

□ **Fluency**: Whether the generated questions are well-posed and natural in terms of grammar and semantic.

□ **Answerable**: Whether the generated questions could be answered by the context paragraph.

□ **Significance**: Whether the questions are focusing on the significant aspect of the paragraph or the trivial one. This criterion could be evaluate based on two standards: (1), whether the generated question is just a simple syntactical transformation of the paragraph sentence (2), whether the corresponding answers are trivial.

We adopt five models for comparison: (a): NQG++ (b): SAQG (c): our model without answer pivot (d) our model. (e) our model with supervised answer pivot training ($\mathcal{L}_S$ in Eq. 12). We shuffle the questions generated by the different models before the assessment. Ratings were collected on a 1-to-5 scale. The result is shown in Table 2.

We can see from the table that the fluency of our model and SAQG are superior to NQG++. As the building block of our model and SAQG are both based on Transformer (Vaswani et al. 2017), which has already shown advantage in

[2] https://github.com/magic282/NQG

[3] https://github.com/davidgolub/QuestionGeneration

[4] https://github.com/ujjax/question-generation

[5] https://github.com/yanghoonkim/NQG_ASs2s

	Fluency	Answerable	Significance	Ave.
NQG++	4.02	**4.64**	2.28	3.65
SAQG	4.22	3.26	3.68*	3.72
- answer pivot	4.18	3.20	3.52	3.63
our model	**4.26**	3.98	3.68*	3.97
+ $\mathcal{L}_S$	4.22	4.58	3.62	**4.14**

Table 2: Human assessment of the generated questions.

natural text generation (Radford et al. ; 2019). The NQG++ is an answer-aware model that most of the generated questions are answerable. But the model relies too heavily on the golden answers so the significance score is poor. On the other hand, the answer-agnostic model could generate more significant questions. However, as the answers are not taken into account, some questions are invalid based on the current paragraph, which result in a poor answerable score. Our proposed model takes advantage of both answer-aware and answer-agnostic NQG and achieves a better average score. Besides, the model behaves even better when we incorporate the supervised answer pivot training signal, as it provides more accurate answer position.

In Table 3, we report a few sample outputs of the different models. We found that the LSTM based models sometimes suffer from the over-generation problem (Tu et al. 2016). In addition, we found that adding answer information making the question more specific, and the model without answer information is more general, this may explain the fact that the output of the SAQG and the model without answer pivot is relatively short.

Induced Answer Pivot Analysis

The core of the proposed model is the hidden answer pivot, which would guide the questions to focus on specific answers. In this section, we evaluate the quality of the induced answer pivot. Concretely, after answer pivot prediction in Eq. 3, for each sample paragraph with length n, we have n binary predictions $Z \in \{0, 1\}^n$, each item denotes whether the current word is in the answers. And the ground truth binary labels $A \in \{0, 1\}^n$. Then for each samples, we can define the *precision*, *recall* and *F1* score:

$$\begin{aligned} Precision &= \frac{|A \cap Z \in \mathbf{1}|}{|Z \in \mathbf{1}|} \\ Recall &= \frac{|A \cap Z \in \mathbf{1}|}{|A \in \mathbf{1}|} \\ F1 &= \frac{2 \times Precision \times Recall}{Precision + Recall} \end{aligned} \tag{14}$$

where $|A \cap Z \in \mathbf{1}|$ denotes the number of 1s occur in both A and Z. These metrics are widely used in information retrieval (Manning, Raghavan, and Schütze 2010) and a higher score indicating higher answer accuracy. We compare our model with two baselines.

- **Random**: We randomly predict the binary label for each token with Bernoulli distribution probability 0.5.

Context: *In 1648 before the term genocide had been coined, the Peace of Westphalia was established to protect ethnic, national, racial and in some instances religious groups.*

Human: What year was the Peace of Westphalia signed?

NQG++: what did the peace of westphalia established established established ...

SAQG: when did the peace of westphalia established ?

-AP: when did the term been coined ?

Our model: why does the peace of westphalia established ?

$+\mathcal{L}_S$**:** what does the peace of westphalia protect ?

Context: *In 1507, he was ordained to the priesthood, and in 1508, von Staupitz, first dean of the newly founded University of Wittenberg, sent for Luther, to teach theology. He received a bachelor's degree in Biblical studies on 9 March 1508, and another bachelor's degree in the Sentences by Peter Lombard in 1509.*

Human: When was Martin Luther ordained as a priest?

NQG++: when did he received a bachelor's degree in biblical studies ?

SAQG: who is von staupitz ?

-AP: what did he teach ?

Our model: who sent luther to teach theology ?

$+\mathcal{L}_S$**:** who is the first dean of the newly founded university of wittenberg ?

Context: *For exercise, Tesla walked between 8 to 10 miles per day. He squished his toes one hundred times for each foot every night, saying that it stimulated his brain cells.*

Human: What was the daily distance walked by Tesla?

NQG++: who walked between 8 to 10 miles a day ?

SAQG: why did he walk every night ?

-AP: how much miles did tesla walked ?

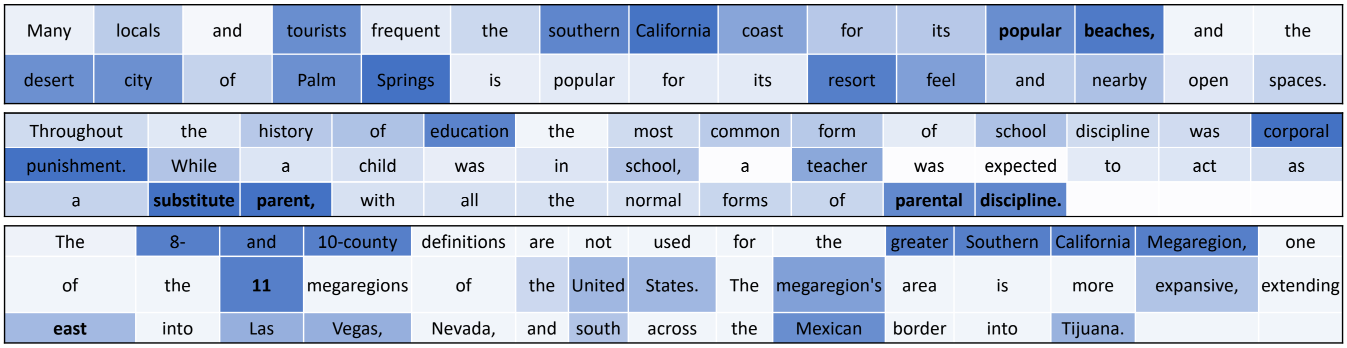
Our model: why did he walked every night ?

$+\mathcal{L}_S$**:** why did tesla walked every night ?

Table 3: Some generated questions on SQuAD dev set.

- **SynNet**: It contains an LSTM based IOB tagging model. We process the label with B and I to 1 and O to zero.
- **SeqLabel**: We remove the question generation of our model and only optimize Eq. 12. Thus, our model is reduced to a pure sequence labeling model with only answer position supervision.

The result is shown in Table 4. It is clear if the models are supervised by the ground truth labels, the answer prediction accuracy would be improved a lot. However, our model has a promising result even if it was trained without explicit answer supervision, which demonstrates that modeling the answer information during question generation is useful. Just like the attention (Bahdanau, Cho, and Bengio 2014) mech-



Many	locals	and	tourists	frequent	the	southern	California	coast	for	its	**popular**	**beaches,**	and	the	
desert	city	of	Palm	Springs	is	popular	for	its	resort	feel	and	nearby	open	spaces.	

Throughout	the	history	of	education	the	most	common	form	of	school	discipline	was	corporal
punishment.	While	a	child	was	in	school,	a	teacher	was	expected	to	act	as
a	**substitute**	**parent,**	with	all	the	normal	forms	of	**parental**	**discipline.**			

The	8-	and	10-county	definitions	are	not	used	for	the	greater	Southern	California	Megaregion,	one
of	the	**11**	megaregions	of	the	United	States.	The	megaregion's	area	is	more	expansive,	extending
east	into	Las	Vegas,	Nevada,	and	south	across	the	Mexican	border	into	Tijuana.		

Figure 3: The generated answer pivot of our model. Darker cells means the higher probability of the word to be the answer. We use the bold words to denote the ground truth answers.

	Precision	Recall	F1
Random	6.4	51.3	21.9
SynNet	76.4	84.3	81.9
SeqLabel	72.9	88.6	80.5
- answer pivot	19.3	59.6	39.4
our model	58.6	73.4	66.2
$+\mathcal{L}_S$	**77.4**	**89.6**	**83.9**

Table 4: The precision, recall, F1 of the generated answers.

anism, this hidden pivot is learned from data and act as a pivot to guide the subsequence generation.

In Figure 3, we plot some sampled answer pivot based on g in Eq. 3. We can see that our model is generally focusing on the significant part of the paragraph and pay less attention to the stop words that are unlikely to be the answers. This interesting result also sheds light on a promising unsupervised machine comprehension task, where we could obtain the answers based on the abundant paragraph-question pairs. We leave this for future study.

Transfer Learning on Machine Comprehension

Since one of the most important applications of NQG is generating more training data for MRC. In this section, we use the proposed models to generate more data for SQuAD. Concretely, we use the 2019-01-01 wikidumps[6] with WikiExtractor[7] tools. We extract only the text passages and ignore lists, tables, and headers, then we randomly sampled several paragraphs from the dumps that do not appear in the SQuAD datasets.

We compare our model with two baselines. (a): NQG++: we use a simple sequence tagging model to generate the candidate answers and then use NQG++ to generate the questions. (b): SynNet: it is a joint model that the answers and questions are generated simultaneously. We use Bert (Devlin et al. 2018) base model as the MRC model. We gradually increase the number of training data from different NQG and train Bert with these additional data.

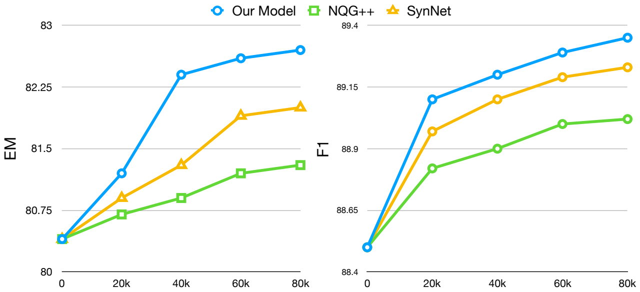


Figure 4: The SQuAD result of Bert base model. The x-axis is the number of additional data we fed to the model.

We evaluate the MRC result in terms of exact match (EM) and F1. The results are shown in Figure 4. We can see that the data generated by our model are more beneficial to the MRC. We found that the answers generated by SynNet and the sequence tagging model are most named entities, and sometimes the questions generated on them are very simple, usually a transformation of the paragraph text, which has less contribution to improve the MRC quality. The results of MRC further affirm that taking question information to infer the candidate answer, which is model by the answer pivot in this work, is important for NQG.

Conclusion

We introduce a novel hidden answer pivot module to the neural question generation which explicitly modeling the hidden answer information. It takes advantage of the previous answer-aware and answers agnostic NQG to generate non-trivial and answerable questions. We introduce the straight through estimator to optimize the model. Experimental results demonstrate the advantage of the proposed model in terms of the quality of the generated questions and the induced answers. Furthermore, the generated QA pairs also improve the downstream MRC task.

Acknowledgments

We thank the anonymous reviewers for their helpful comments.

[6] https://dumps.wikimedia.org/enwiki/

[7] https://github.com/attardi/wikiextractor

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Chali, Y., and Hasan, S. A. 2015. Towards topic-to-question generation. *Computational Linguistics* 41(1):1–20.

Chen, Y.; Wu, L.; and Zaki, M. J. 2019. Reinforcement learning based graph-to-sequence model for natural question generation. *arXiv preprint arXiv:1908.04942*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dong, X.; Hong, Y.; Chen, X.; Li, W.; Zhang, M.; and Zhu, Q. 2018. Neural question generation with semantics of question type. In *NLPCC*, 213–223. Springer.

Du, X.; Shao, J.; and Cardie, C. 2017. Learning to ask: Neural question generation for reading comprehension. 1342–1352.

Duan, N.; Tang, D.; Chen, P.; and Zhou, M. 2017. Question generation for question answering. In *EMNLP*, 866–874.

Golub, D.; Huang, P.-S.; He, X.; and Deng, L. 2017. Two-stage synthesis networks for transfer learning in machine comprehension. *arXiv preprint arXiv:1706.09789*.

Heilman, M., and Smith, N. A. 2010. Good question! statistical ranking for question generation. In *HLT-NAACL*.

Heilman, M. 2011. Automatic factual question generation from text. *Language Technologies Institute School of Computer Science Carnegie Mellon University* 195.

Hinton, G.; Srivastava, N.; and Swersky, K. 2012. Neural networks for machine learning. *Coursera, video lectures* 264.

Hosking, T., and Riedel, S. 2019. Evaluating rewards for question generation models. In *NAACL*, 2278–2283.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks. In *NIPS*.

Hussein, H.; Elmogy, M.; and Guirguis, S. 2014. Automatic english question generation system based on template driven scheme. *IJCSI* 11(6):45.

Kim, Y.; Lee, H.; Shin, J.; and Jung, K. 2019. Improving neural question generation using answer separation. In *AAAI*, volume 33, 6602–6609.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *ICLR*.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *CoRR* abs/1312.6114.

Kudo, T., and Richardson, J. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Kumar, V.; Ramakrishnan, G.; and Li, Y.-F. 2018. A framework for automatic question generation from text using deep reinforcement learning. *arXiv preprint arXiv:1808.04961*.

Liu, L.; Utiyama, M.; Finch, A.; and Sumita, E. 2016. Neural machine translation with supervised attention. In *COLING*.

Lucas, J.; Tucker, G.; Grosse, R.; and Norouzi, M. 2019. Understanding posterior collapse in generative latent variable models.

Manning, C.; Raghavan, P.; and Schütze, H. 2010. Introduction to information retrieval. *Natural Language Engineering* 16(1).

Mostow, J., and Chen, W. 2009. Generating instruction automatically for the reading strategy of self-questioning. In *AIED*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. Improving language understanding by generative pre-training.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Rus, V.; Wyse, B.; Piwek, P.; Lintean, M.; Stoyanchev, S.; and Moldovan, C. 2010. The first question generation shared task evaluation challenge.

Scialom, T.; Piwowarski, B.; and Staiano, J. 2019. Self-attention architectures for answer-agnostic neural question generation. In *ACL*, 6027–6032.

Shen, D.; Su, Q.; Chapfuwa, P.; Wang, W.; Wang, G.; Henao, R.; and Carin, L. 2018. Nash: Toward end-to-end neural architecture for generative semantic hashing. In *ACL*.

Song, L.; Wang, Z.; Hamza, W.; Zhang, Y.; and Gildea, D. 2018. Leveraging context information for natural question generation. In *NAACL*, 569–574.

Song, L.; Wang, Z.; and Hamza, W. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058*.

Subramanian, S.; Wang, T.; Yuan, X.; Zhang, S.; Trischler, A.; and Bengio, Y. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, 78–88.

Sun, X.; Liu, J.; Lyu, Y.; He, W.; Ma, Y.; and Wang, S. 2018. Answer-focused and position-aware neural question generation. In *EMNLP*, 3930–3939.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, 1057–1063.

Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; and Li, H. 2016. Modeling coverage for neural machine translation. In *ACL*, 76–85.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, 5998–6008.

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

Xiao, H.; Wang, F.; Feng, Y.; and Zheng, J. 2018. Dual ask-answer network for machine reading comprehension. *arXiv preprint arXiv:1809.01997*.

Yuan, X.; Wang, T.; Gulcehre, C.; Sordoni, A.; Bachman, P.; Subramanian, S.; Zhang, S.; and Trischler, A. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012*.

Zhou, Q.; Yang, N.; Wei, F.; Tan, C.; Bao, H.; and Zhou, M. 2017. Neural question generation from text: A preliminary study. In *NLPCC*.

Çaglar Gülçehre; Ahn, S.; Nallapati, R.; Zhou, B.; and Bengio, Y. 2016. Pointing the unknown words. *ArXiv* abs/1603.08148.