Generating Questions for Knowledge Bases via Incorporating Diversified Contexts and Answer-Aware Loss

Cao Liu[1,2], **Kang Liu**[1,2], **Shizhu He**[1,2], **Zaiqing Nie**[3], **Jun Zhao**[1,2]
[1] National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, 100190, China
[2] University of Chinese Academy of Sciences, Beijing, 100049, China
[3] Alibaba AI Labs, Beijing, 100029, China
{cao.liu, kliu, shizhu.he, jzhao}@nlpr.ia.ac.cn
zaiqing.nzq@alibaba-inc.com

Abstract

We tackle the task of question generation over knowledge bases. Conventional methods for this task neglect two crucial research issues: 1) the given predicate needs to be expressed; 2) the answer to the generated question needs to be definitive. In this paper, we strive toward the above two issues via incorporating diversified contexts and answer-aware loss. Specifically, we propose a neural encoder-decoder model with multi-level copy mechanisms to generate such questions. Furthermore, the answer aware loss is introduced to make generated questions corresponding to more definitive answers. Experiments demonstrate that our model achieves state-of-the-art performance. Meanwhile, such generated question can express the given predicate and correspond to a definitive answer.

1 Introduction

Question Generation over Knowledge Bases (KBQG) aims at generating natural language questions for the corresponding facts on KBs, and it can benefit some real applications. Firstly, KBQG can automatically annotate question answering (QA) datasets. Secondly, the generated questions and answers will be able to augment the training data for QA systems. More importantly, KBQG can improve the ability of machines to actively ask questions on human-machine conversations (Duan et al., 2017; Sun et al., 2018). Therefore, this task has attracted more attention in recent years (Serban et al., 2016; Elsahar et al., 2018).

Specifically, KBQG is the task of generating natural language questions according to the input facts from a knowledge base with triplet form, like <subject, predicate, object>. For example, as illustrated in Figure 1, KBQG aims at generating a question "Which city is Statue of Liberty located in?" (Q3) for the input factual triplet

Input	<Statue of Liberty, location/containedby, ***New York City***>		
Output	Matching Predicate	Definite Answer	Question
Q1	×	-	Who **created** the Statue of Liberty?
Q2	√	×	**Where** is Statue of Liberty in?
Q3	√	√	Which ***city*** is Statue of Liberty **located** in?

Figure 1: Examples of KBQG. We aims at generating questions like Q3 which expresses (matches) the given predicate and refers to a definitive answer.

"<Statue of Liberty, location/containedby[1], New York City>". Here, the generated question is associated to the subject "*Statue of Liberty*" and the predicate `fb:location/containedby`) of the input fact, and the answer corresponds to the object "*New York City*".

As depicted by Serban et al. (2016), KBQG is required to transduce the triplet fact into a question about the subject and predicate, where the object is the correct answer. Therefore, it is a key issue for KBQG to correctly understand the knowledge symbols (subject, predicate and object in the triplet fact) and then generate corresponding text descriptions. More recently, some researches have striven toward this task, where the behind intuition is to construct implicit associations between facts and texts. Specifically, Serban et al. (2016) designed an encoder-decoder architecture to generate questions from structured triplet facts. In order to improve the generalization for KBQG, Elsahar et al. (2018) utilized extra contexts as input via distant supervisions (Mintz et al., 2009), then a decoder is equipped with attention and part-of-speech (POS) copy mechanism to generate questions. Finally, this model obtained significant improvements. Nevertheless, we observe that there are still two important research issues (RIs) which are not processed well or even neglected.

[1]We omit the domain of the predicate for sake of brevity.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 2431–2441,
Hong Kong, China, November 3–7, 2019.

RI-1: *The generated question is required to* ***express the given predicate*** *in the fact*. For example in Figure 1, Q1 does not express (match) the predicate (`fb:location/containedby`) while it is expressed in Q2 and Q3. Previous work (Elsahar et al., 2018) usually obtained predicate textual contexts through distant supervision. However, the distant supervision is noisy or even wrong (e.g. "X is the husband of Y" is the relational pattern for the predicate `fb:marriage/spouse`, so it is wrong when "X" is a woman). Furthermore, many predicates in the KB have no predicate contexts. We make statistic in the resources released by Elsahar et al. (2018), and find that only 44% predicates have predicate textual context[2]. Therefore, it is prone to generate error questions from such without-context predicates.

RI-2: *The generated question is required to contain a* ***definitive answer***. A definitive answer means that one question only associates with a determinate answer rather than alternative answers. As an example in Figure 1, Q2 may contain ambiguous answers since it does not express the refined answer type. As a result, different answers including "*United State*", "*New York City*", etc. may be correct. In contrast, Q3 refers to a definitive answer (the object "*New York City*" in the given fact) by restraining the answer type to a city. We believe that Q3, which expresses the given predicate and refers to a definitive answer, is a better question than Q1 and Q2. In previous work, Elsahar et al. (2018) only regarded a most frequently mentioned entity type as the textual context for the subject or object in the triplet. In fact, most answer entities have multiple types, where the most frequently mentioned type tends to be universal (e.g. a broad type "administrative region" rather than a refined type "US state" for the entity "*New York*"). Therefore, generated questions from Elsahar et al. (2018) may be difficult to contain definitive answers.

To address the aforementioned two issues, we exploit more **diversified contexts** for the given facts as textual contexts in an encoder-decoder model. Specifically, besides using predicate contexts from the distant supervision utilized by Elsahar et al. (2018), we further leverage the domain, range and even topic for the given predicate as contexts, which are off-the-shelf in KBs (e.g. the range and the topic for the predicate `fb:location/containedby` are "location" and "containedby", respectively[1]). Therefore, 100% predicates (rather than 44%[2] of those in Elsahar et al.) have contexts. Furthermore, in addition to the most frequently mentioned entity type as contexts used by Elsahar et al. (2018), we leverage the type that best describes the entity as contexts (e.g. a refined entity type[3] "US state" combines a broad type "administrative region" for the entity "New York"), which is helpful to refine the entity information. Finally, in order to make full use of these contexts, we propose context-augmented fact encoder and multi-level copy mechanism (KB copy and context copy) to integrate diversified contexts, where the multi-level copy mechanism can copy from KB and textual contexts simultaneously. For the purpose of further making generated questions correspond to definitive answers, we propose the **answer-aware loss** by optimizing the cross-entropy between the generated question and answer type words, which is beneficial to generate precise questions.

We conduct experiments on an open public dataset. Experimental results demonstrate that the proposed model using diversified textual contexts outperforms strong baselines (+4.5 BLEU4 score). Besides, it can further increase the BLEU score (+5.16 BLEU4 score) and produce questions associated with more definitive answers by incorporating answer-aware loss. Human evaluations complement that our model can express the given predicate more precisely.

In brief, our main contributions are as follows:

(1) We leverage diversified contexts and multi-level copy mechanism to alleviate the issue of incorrect predicate expression in traditional methods.

(2) We propose an answer-aware loss to tackle the issue that conventional methods can not generate questions with definitive answers.

(3) Experiments demonstrate that our model achieves state-of-the-art performance. Meanwhile, such generated question can express the given predicate and refer to a definitive answer.

2 Task Description

We leverage textual contexts concerned with the triplet fact to generate questions over KBs. The

[2]We map the "prop_text_evidence.csv" file to the "property.vocab" file in Elsahar et al. (2018)

[3]We obtain such representative entity types through the predicate `fb:topic/notable_types` in freebase.

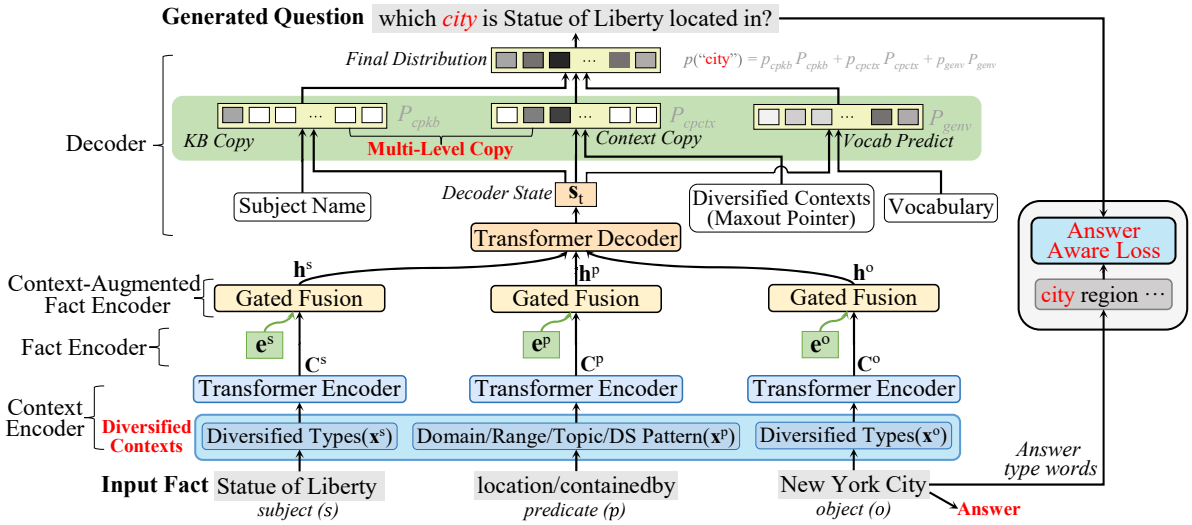

Figure 2: Overall structure of the proposed model for KBQG. A **context encoder** is firstly employed to encode each textual context (Sec. 3.1), where "Diversified Types" represents the subject (object) context, and "DS pattern" denotes the relational pattern from distant supervisions. At the same time, a **fact encoder** transforms the fact into low-dimensional representations (Sec. 3.2). The above two encoders are aggregated by the **context-augmented fact encoder** (Sec. 3.3). Finally, the aggregated representations are fed to the **decoder** (Sec. 3.4), where the decoder leverages multi-level copy mechanism (KB copy and context copy) to generate target question words.

task of KBQG can be formalized as follows:

$$P(Y|F) = \prod_{t=1}^{|Y|} P(y_t|y_{<t}, F, C) \quad (1)$$

where $F = (s, p, o)$ represents the subject (s), predicate (p) and object (o) of the input triplet, $C = \{\mathbf{x}^s, \mathbf{x}^p, \mathbf{x}^o\}$ denotes a set of additional textual contexts, $Y = (y_1, y_2, ..., y_{|Y|})$ is the generated question, $y_{<t}$ represents all previously generated question words before time step t.

3 Methodology

Our model extends the encoder-decoder architecture (Cho et al., 2014b) with three encoding modules and two copy mechanisms in the decoder. The model overview is shown in Figure 2 along with its caption. It should be emphasized that we additionally design an answer-aware loss to make the generated question associated with a definitive answer (Sec. 3.5.2).

3.1 Context Encoder

Inspired by the great success of transformer (Vaswani et al., 2017) in sequence modeling (Shen et al., 2018), we adopt a transformer encoder to encode each textual context separately. Take the subject context $\mathbf{x}^s$ as an example, $\mathbf{x}^s = (x_1^s, x_2^s, ..., x_{|s|}^s)$ is concatenated from diversified types for the subject, and x_i^s is the i-th token in the subject context, $|s|$ stands for the length of the subject context. Firstly, $\mathbf{x}^s$ is mapped into a query matrix $\mathbf{Q}$, where $\mathbf{Q}$ is constructed by summing the corresponding token embeddings and segment embeddings. Similar to BERT (Devlin et al., 2019), segment embeddings are the same for tokens of $\mathbf{x}^s$ but different for that of $\mathbf{x}^p$ (predicate context) or $\mathbf{x}^o$ (object context). Based on the query matrix, transformer encoder works as follows:

$$\mathbf{Q}_j = \mathbf{Q}\mathbf{W}_j^Q, \mathbf{K}_j = \mathbf{K}\mathbf{W}_j^K, \mathbf{V}_j = \mathbf{V}\mathbf{W}_j^V \quad (2)$$

$$head_j = \text{softmax}(\mathbf{Q}_j\mathbf{K}_j^T/\sqrt{d/h})\mathbf{V}_j \quad (3)$$

$$\mathbf{H}^s = \text{Concat}(head_1, head_2, ..., head_h)\mathbf{W}_0 \quad (4)$$

$$\mathbf{N}^s = \text{LayerNorm}(\mathbf{Q} + \mathbf{H}^s) \quad (5)$$

$$\mathbf{C}^s = \max(0, \mathbf{N}^s\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2 \quad (6)$$

where $\mathbf{K}$ and $\mathbf{V}$ are the key matrix and value matrix, respectively. It is called self-attention because $\mathbf{K}$ and $\mathbf{V}$ are equal to the query matrix $\mathbf{Q} \in \mathbb{R}^{|s|,d}$ in the encoding stage, where d represents the number of hidden units. And h denotes the number of the heads in multi-head attention mechanism of the transformer encoder. It first projects the input matrixes ($\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$) into subspaces h times mapped by different linear projections $\mathbf{W}_j^Q$, $\mathbf{W}_j^K$, $\mathbf{W}_j^V \in \mathbb{R}^{|s|,d/h}$ ($j = 1, 2, ..., h$) in Equation 2. And then h projections perform the scaled dot-product attention to obtain the representation of each head in parallel (Equation 3). Representa-

tions for all parallel heads are concatenated together in Equation 4. After residual connection, layer normalization (Equation 5) and feed forward operation (Equation 6), we can obtain the subject context matrix $\mathbf{C}^s = \{\mathbf{c}_1^s, \mathbf{c}_2^s, ..., \mathbf{c}_{|s|}^s\} \in \mathbb{R}^{|s|,d}$.

Similarly, $\mathbf{C}^p$ and $\mathbf{C}^o$ are obtained from the same transformer encoder for the predicate and object, respectively.

3.2 Fact Encoder

In contrast to general Sequence-to-Sequence (Seq2Seq) model (Sutskever et al., 2014), the input fact is not a word sequence but instead a structured triplet $F = (s, p, o)$. We employ a fact encoder to transform each atom in the fact into a fixed embedding, and the embedding is obtained from a KB embedding matrix. For example, the subject embedding $\mathbf{e}^s \in \mathbb{R}^d$ is looked up from the KB embedding matrix $\mathbf{E}_f \in \mathbb{R}^{k,d}$, where k represents the size of KB vocabulary, and the size of KB embedding is equal to the number of hidden units (d) in Equation 3. Similarly, the predicate embedding $\mathbf{e}^p$ and the object embedding $\mathbf{e}^o$ are mapped from the KB embedding matrix $\mathbf{E}_f$, where $\mathbf{E}_f$ is pre-trained using *TransE* (Bordes et al., 2013) to capture much more fact information in previous work (Elsahar et al., 2018). In our model, $\mathbf{E}_f$ can be pre-trained or randomly initiated (Details in Sec. 4.7.1).

3.3 Context-Augmented Fact Encoder

In order to combine both the context encoder information and the fact encoder information, we propose a context-augmented fact encoder which applies the gated fusion unit (Gong and Bowman, 2018) to integrate the context matrix and the fact embedding. For example, the subject context matrix $\mathbf{C}^s = \{\mathbf{c}_1^s, \mathbf{c}_2^s, ..., \mathbf{c}_{|s|}^s\}$ and the subject embedding vector $\mathbf{e}^s$ are integrated by the following gated fusion:

$$\mathbf{f} = \tanh(\mathbf{W}_f[\mathbf{c}^s, \mathbf{e}^s]) \quad (7)$$

$$\mathbf{g} = \text{sigmoid}(\mathbf{W}_g[\mathbf{c}^s, \mathbf{e}^s]) \quad (8)$$

$$\mathbf{h}^s = \mathbf{g} \odot \mathbf{f} + (1 - \mathbf{g}) \odot \mathbf{e}^s \quad (9)$$

where $\mathbf{c}^s$ is an attentive vector from $\mathbf{e}^s$ to $\mathbf{C}^s$, which is similar to Zhao et al. (2018). The attentive vector $\mathbf{c}^s$ is combined with original subject embedding $\mathbf{e}^s$ as a new enhanced representation $\mathbf{f}$ (Equation 7). And then a learnable gate vector, $\mathbf{g}$ (Equation 8), controls the information from $\mathbf{c}^s$ and $\mathbf{e}^s$ to the final augmented subject vector $\mathbf{h}^s \in \mathbb{R}^d$ (Equation 9), where $\odot$ denotes the element-wise multiplication. Similarly, the augmented predicate vector $\mathbf{h}^p$ and the augmented object vector $\mathbf{h}^o$ are calculated in the same way. Finally, the context-augmented fact representation $\mathbf{H}_f \in \mathbb{R}^{3,d}$ is the concatenation of augmented vectors as follows:

$$\mathbf{H}_f = [\mathbf{h}^s; \mathbf{h}^p; \mathbf{h}^o] \quad (10)$$

3.4 Decoder

The decoder aims at generating a question word sequence. As shown in Figure 2, we also exploit the transformer as the basic block in our decoder. Then we use a multi-level copy mechanism (KB copy and context copy), which allows copying from KBs and textual contexts.

Specifically, we first map the input of the decoder into an embedding representation by looking up word embedding matrix, then we use position embedding (Vaswani et al., 2017) to enhance sequential information. Compared with the transformer encoder in Sec. 3.1, transformer decoder has an extra sub-layer: a fact multi-head attention layer, which is similar to Equation 2-6, where the query matrix is initiated with previous decoder sub-layer while both the key matrix and the value matrix are the augmented fact representation $\mathbf{H}_f$. After feedforward and multiple transformer layers, we obtain the decoder state $\mathbf{s}_t$ at time step t, and then $\mathbf{s}_t$ could be leveraged to generate the target question sequence word by word.

As depicted in Figure 2, we propose multi-level copy mechanism to generate question words. At each time step t, given decoder state $\mathbf{s}_t$ together with input fact F, textual contexts C and vocabulary V, the probabilistic function for generating any target question word y_t is calculated as:

$$P(y_t|\mathbf{s}_t, y_{t-1}, F, C) = p_{genv} P_{genv}(y_t|\mathbf{s}_t, V) + p_{cpkb} P_{cpkb}(y_t|\mathbf{s}_t, F) + p_{cpctx} P_{cpctx}(y_t|\mathbf{s}_t, C) \quad (11)$$

$$p_{genv}, p_{cpkb}, p_{cpctx} = \text{softmax}([\mathbf{s}_t, \mathbf{y}_{t-1}]) \quad (12)$$

where $genv$, $cpkb$ and $cpctx$ denote the vocab generation mode, the KB copy mode and the context copy mode, respectively. In order to control the balance among different modes, we employ a 3-dimensional switch probability in Equation 12, where $\mathbf{y}_{t-1}$ is the embedding of previous generated word, $P_\cdot(\cdot|\cdot)$ indicates the probabilistic score function for generated target word of each mode. In the three probability score functions, P_{vocab} is typically performed by a $softmax$ classifier over

a fixed vocabulary V based on the word embedding similarity, and the details of P_{cpkb} and P_{cpctx} are in the following.

3.4.1 KB Copy

Previous study found that most questions contain the subject name or its aligns in SimpleQuestion (Petrochuk and Zettlemoyer, 2018). However, the predicate name and object name hardly appear in the question. Therefore, we only copy the subject name in the KB copy, where $P_{cpkb}(y_t|\mathbf{s}_t, f)$, the probability of copying the subject name, is calculated by a neural network function with a multilayer perceptron (MLP) projected from $\mathbf{s}_t$.

3.4.2 Context Copy

Elsahar et al. (2018) demonstrated the effectiveness of POS copy for the context. However, such a copy mechanism heavily relies on POS tagging. Inspired by the CopyNet (Gu et al., 2016), we directly copy words in the textual contexts C, and it does not rely on any POS tagging. Specifically, the input sequence χ for the context copy is the concatenation of all words in the textual contexts C. Unfortunately, χ is prone to contain repeated words because it consists of rich contexts for subject, predicate and object. The repeated words in the input sequence tend to cause repetition problems in output sequences (Tu et al., 2016). We adopt the maxout pointer (Zhao et al., 2018) to address the repetition problem. Instead of summing all the probabilistic scores for repeated input words, we limit the probabilistic score of repeated words to their maximum score as Equation 13:

$$P_{cpctx}(y_t|.)=\begin{cases}\max\limits_{m,\, where\ \chi_m=y_t} sc_{t,m} & y_t \in \chi \\ 0 & otherwise\end{cases} \quad (13)$$

where χ_m represents the m-th token in the input context sequence χ, $sc_{t,m}$ is the probabilistic score of generating the token χ_m at time step t, and $sc_{t,m}$ is calculated by a softmax function over χ.

3.5 Learning

3.5.1 Question-Aware Loss

It is totally differential for our model to obtain question words, and it can be optimized in an end-to-end manner by back-propagation. Given the input fact F, additional textual context C and target question word sequence Y, the object function is to optimize the following negative log-likelihood:

$$\mathcal{L}_{ques\_loss} = \frac{-1}{|Y|}\sum_{t=1}^{|Y|}\log[P(y_t|\mathbf{s}_t, y_{t-1}, F, C)] \quad (14)$$

The question-aware loss $\mathcal{L}_{ques\_loss}$ does not require any additional labels to optimize because the three modes share a same softmax classifier to keep a balance (Equation 12), and they can learn to coordinate each other by minimizing $\mathcal{L}_{ques\_loss}$.

3.5.2 Answer-Aware loss

It is able to generate questions similar to the labeled questions by optimizing the question-aware loss $\mathcal{L}_{ques\_loss}$. However, there is an ambiguous problem in the annotated questions where the questions have alternative answers rather than determinate answers (Petrochuk and Zettlemoyer, 2018). In order to make generated questions correspond to definitive answers, we propose a novel answer-aware loss. By answer-aware loss, we aim at generating an answer type word in the question, which contributes to generating a question word matching the answer type. Formally, the answer-aware loss is in the following:

$$\mathcal{L}_{ans\_loss} = \min_{a_n, a_n \in A} \min_{y_t, y_t \in Y} H_{a_n, y_t} \quad (15)$$

where $A = \{a_n\}_{n=1}^{|A|}$ is a set of answer type words. We treat object type words as the answer type words because the object is the answer. H_{a_n,y_t} denotes the cross entropy between the answer type word a_n and the generated question word y_t. Finally, the minimum cross entropy is regarded as the answer-aware loss $\mathcal{L}_{ans\_loss}$. Optimizing $\mathcal{L}_{ans\_loss}$ means that the model aims at generating an answer type word in the generated question sequence. For example, the model tends to generate Q3 rather than Q2 in Figure 1, because Q3 contains an answer type word—"city". Similarly, $\mathcal{L}_{ans\_loss}$ could be optimized in an end-to-end manner, and it can integrate $\mathcal{L}_{ques\_loss}$ by a weight coefficient λ to the total loss as follows:

$$\mathcal{L}_{total\_loss} = \mathcal{L}_{ques\_loss} + \lambda\mathcal{L}_{ans\_loss} \quad (16)$$

4 Experiment

4.1 Experimental Settings

4.1.1 Experimental Data Details

We conduct experiments on the SimpleQuestion dataset (Bordes et al., 2015), and there

are 75910/10845/21687 question answering pairs (QA-pairs) for training/validation/test. In order to obtain **diversified contexts**, we additionally employ domain, range and topic of the predicate to improve the coverage of predicate contexts. In this way, 100% predicates (rather than 44%[2] of those in Elsahar et al.) have contexts. For the subject and object context, we combine the most frequently mentioned entity type (Elsahar et al., 2018) with the type that best describe the entity[3]. The KB copy needs subject names as the copy source, and we map entities with their names similar to those in Mohammed et al. (2018). The data details are in Appendix A and submitted Supplementary Data.

4.1.2 Evaluation Metrics

Following (Serban et al., 2016; Elsahar et al., 2018), we adopt some word-overlap based metrics (WBMs) for natural language generation including BLEU-4 (Papineni et al., 2002), ROUGE_L (Lin, 2004) and METEOR (Denkowski and Lavie, 2014). However, such metrics still suffer from some limitations (Novikova et al., 2017). Crucially, it might be difficult for them to measure whether generated questions that express the given predicate and refer to definitive answers. To better evaluate generated questions, we run two further evaluations as follows.

(1) ***Predicate identification***: Following Mohammed et al. (2018), we employ annotators to judge whether the generated question expresses the given predicate in the fact or not. The score for predicate identification is the percentage of generated questions that express the given predicate.

(2) ***Answer coverage***: We define a novel metric called answer coverage to identify whether the generated question refers to a definitive answer. Specifically, answer coverage is obtained by automatically calculating the percentage of questions that contain answer type words, and answer type words are object contexts (entity types for the object are regarded as answer type words).

Furthermore, it is hard to automatically evaluate the naturalness of generated questions. Following Mohammed et al. (2018), we adopt human evaluation to measure the naturalness by a score of 0-5.

4.1.3 Comparison with State-of-the-arts

We compare our model with following methods.

(1) *Template*: A baseline in Serban et al. (2016), it randomly chooses a candidate fact F_c in the training data to generate the question, where F_c shares the same predicate with the input fact.

(2) *Serban et al. (2016)*: We compare our methods with the single placeholder model, which performs best in Serban et al. (2016).

(3) *Elsahar et al. (2018)*: We compare our methods with the model utilizing copy actions, the best performing model in Elsahar et al. (2018). Although this model is designed to a zero-shot setting (for unseen predicates and entity type), it has good abilities to generate better questions (on known or unknown predicates and entity types) represented in the additional context input and SPO copy mechanism.

4.1.4 Implementation Details

To make our model comparable to the comparison methods, we keep most parameter values the same as Elsahar et al. (2018). We utilize RMSProp algorithm with a decreasing learning rate (0.001), batch size (200) to optimize the model. The size of KB embeddings is 200, and KB embeddings are pre-trained by TransE (Bordes et al., 2013). The word embeddings are initialized by the pre-trained Glove word vectors[4] with 200 dimensions. In the transformer, we set the hidden units d to 200, and we employ 4 paralleled attention head and a stack of 5 identical layers. We set the weight (λ) of the answer-aware loss to 0.2.

4.2 Overall Comparisons

Model	BLEU4	ROUGE_L	METEOR
Template	31.36	*	33.12
Serban et al. (2016)	33.32	*	35.38
Elsahar et al. (2018)	36.56	58.09	34.41
Our Model	41.09	68.68	47.75
Our $\text{Model}_{\text{ans\_loss}}$	**41.72**	**69.31**	**48.13**

Table 1: Overall comparisons on the test data, where "ans_loss" represents answer-aware loss.

In Table 1, we compare our model with the typical baselines on word-overlap based metrics. It is evident that our model is remarkably better than baselines on all metrics, where the BLEU4 score increases 4.53 compared with the strongest baseline (Elsahar et al., 2018). Especially, incorporating answer-aware loss (the last line in Table 1) further improves the performance (+5.16 BLEU4).

4.3 Performances on Predicate Identification

To evaluate the ability of our model on predicate identification, we sample 100 generated question-

[4]http://nlp.stanford.edu/data/glove.6B.zip

Model	Pred. Identification
Serban et al. (2016)	53.5
Elsahar et al. (2018)	71.5
Our Model$_{\text{ans_loss}}$	**75.5**

Table 2: Performances on predicate identification.

s from each model, and then two annotators are employed to judge whether the generated question expresses the given predicate. The Kappa for inter-annotator statistics is 0.611, and p-value for all scores is less than 0.005. As shown in Table 2, we can see that our model has a significant improvement in the predicate identification.

4.4 Performances on Answer Coverage — The Effectiveness of Answer-Aware Loss

Model	λ	BLEU4	Ans$_{\text{cov}}$
Elsahar et al. (2018)	0	36.56	59.49
Our Model	0	41.09	61.65
Our Model$_{\text{ans_loss}}$	0.05	41.55	62.27
Our Model$_{\text{ans_loss}}$	0.2	**41.72**	64.23
Our Model$_{\text{ans_loss}}$	0.5	41.57	**65.50**
Our Model$_{\text{ans_loss}}$	1.0	41.34	65.25

Table 3: Performances on answer coverage, where "Ans$_{\text{cov}}$" denotes the metric of answer coverage. "λ" is the weight of the answer-aware loss in Equation 16.

Table 3 reports performances on BLUE4 and answer coverage (Ans$_{\text{cov}}$). We can obtain that:

(1) When answer-aware loss is not leveraged ($\lambda = 0$), advantages of performance are obvious in our model. Note that the answer coverage is 55.23 on the human-labeled questions. Although our model does not explicitly capture answer information, it still obtains a high answer coverage, which may be because our diversified contexts contain rich answer type words.

(2) To demonstrate the effectiveness of answer-aware loss, we set the weight of answer-aware loss (λ) to 0.05/0.2/0.5/1.0 (the last four lines in Table 3). It can be seen that our model, incorporating answer-aware loss, has a significant improvement on answer coverage while there is no performance degradation on BLEU4 compared with $\lambda = 0$, which indicates that answer-aware loss contributes to generating better questions. Especially, the generated questions are more precise because they refer to more definitive answers with high Ans$_{\text{cov}}$.

(3) It tends to correspond to alternative answers (object in the triplet fact) for some predicates such as `fb:location/containedby`, while other predicates (e.g. `fb:person/gender`) may refer to a definitive answer. To investigate our model, by incorporating answer-aware loss, does not generate an answer type word in a mandatory way, we found 20.5% predicate corresponds to the generated questions without answer type words when our model obtains the highest Ans$_{\text{cov}}$ (λ=0.5), and it is very close to 21.7% for the one in human-annotated questions. This demonstrates that the answer-aware loss does not force all predicates to generate questions with answer type words.

4.5 Ablation Study

Model	BLEU4	ROUGE$_{\text{L}}$	METEOR
Our Model$_{\text{ans_loss}}$	**41.72**	**69.31**	**48.13**
w/o context copy	41.27	68.36	47.54
w/o KB copy	41.04	68.66	47.72
w/o answer-aware loss	41.09	68.68	47.75
w/o diversified contexts	40.53	68.52	47.66

Table 4: Ablation study by removing the main components, where "w/o" means without, and "w/o diversified contexts" represents that diversified contexts are replaced by contexts used in Elsahar et al. (2018).

In order to validate the effectiveness of model components, we remove some important components in our model, including context copy, KB copy, answer-aware loss and diversified contexts. The results are shown in Table 4. We can see that removing any component brings performance decline on all metrics. It demonstrates that all these components are useful. Specifically, the last line in Table 4, replacing diversified contexts with contexts used in Elsahar et al. (2018), has more obvious performance degradation.

4.6 Performances on Naturalness

Model	Naturalness
Serban et al. (2016)	2.96
Elsahar et al. (2018)	2.23
Our Model$_{\text{ans_loss}}$	**3.56**

Table 5: Performances on naturalness.

Human evaluation is important for generated questions. Following Elsahar et al. (2018), we sample 100 questions from each system, and then two annotators measure the naturalness by a score of 0-5. The Kappa coefficient for inter-annotator is 0.629, and p-value for all scores is less than 0.005. As shown in Table 5, Elsahar et al. (2018) perform poorly on naturalness, while our model obtains the

highest score on naturalness, which demonstrates our model can deliver more natural questions than baselines.

4.7 Discussion

4.7.1 Without Pre-trained KB Embeddings

Model	TransE	BLEU4	ROUGE_L	METEOR
Elsahar et al. (2018)	True	36.56	58.09	34.41
Elsahar et al. (2018)	False	33.67	55.57	33.20
Our $\text{Model}_{\text{ans\_loss}}$	True	41.72	69.31	48.13
Our $\text{Model}_{\text{ans\_loss}}$	False	41.55	68.59	47.52

Table 6: Performances of whether using the pre-trained KB embedding by transE.

Pre-trained KB embeddings may provide rich structured relational information among entities. However, it heavily relies on large-scale triplets, which is time and resource-intensive. To investigate the effectiveness of pre-trained KB embedding for KBQG, we report the performance of KBQG whether using pre-trained KB embeddings by simply applying TransE. Table 6 shows that the performance of KBQG is degraded without TransE embeddings. In comparison, Elsahar et al. (2018) obtain obvious degradation on all metrics while there is only a slight decline in our model. We believe that it may owe to the context-augmented fact encoder since our model drops to 40.87 on the BLEU4 score without context-augmented fact encoder and transE embeddings.

4.7.2 The Effectiveness of Generated Questions for Enhancing Question Answering over Knowledge Bases

Data Type	Accuracy
human-labeled data	68.97
+ gen_data (Serban et al., 2016)	68.53
+ gen_data (Elsahar et al., 2018)	69.13
+ gen_data (Our $\text{Model}_{\text{ans\_loss}}$)	**69.57**

Table 7: Performances of generated questions for QA.

Previous experiments demonstrate that our model can deliver more precise questions. To further prove the effectiveness of our model, we will see how useful the generated questions are for training a question answering system over knowledge bases. Specifically, we combine human-labeled data with the same amount of model-generated data to a typical QA system (Mohammed et al., 2018). The accuracy of QA is shown in Table 7. We can observe that adding generative questions may weaken the performance of QA (drop from 68.97 to 68.53 in Table 7). Our generated questions achieve the best performance on the QA system. It indicates that our model generates more precise question and has improved QA performances greatly.

4.7.3 Speed

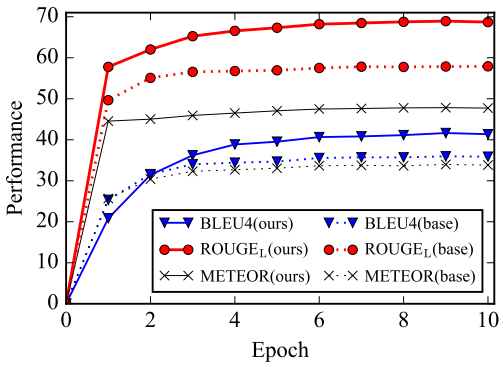

Figure 3: Performance on valid data through epochs, where "base" is the method in Elsahar et al. (2018).

In order to further explore the convergence speed, we plot the performances on valid data through epochs in Figure 3. Our model has much more information to learn, and it may have a bad impact on the convergence speed. Nevertheless, our model can copy KB elements and textual context simultaneously, which may accelerate the convergence speed. As demonstrated in Figure 3, our model achieves the best performances on almost epochs. After about 6 epochs, performances on our model become stable and convergent.

4.7.4 Case Study

Figure 4 lists referenced question and generated questions by different models. It can be seen that our generated questions can better express the target predicate such as ID 1 (marked as underline). In ID 2, although all questions express the target predicate correctly, only our question refers to a definitive answer since it contains an answer type word "city" (marked as **bold**). It should be emphasized that the questions, generated by our method with answer-aware loss, do not always contain answer type words (ID 1 and 3).

5 Related Work

Our work is inspired by a large number of successful applications using neural encoder-decoder

ID	Model	Question
1	Reference	what is the origin of *Kate Bush ?*
	Serban et al.	where is catherine bush buried ?
	Elsahar et al.	what is the artist of catherine bush ?
	Ours	what is the origin of the artist *Kate Bush* ?
2	Reference	what area contains *River Yare* ?
	Serban et al.	where is the *River Yare* ?
	Elsahar et al.	where is the *River Yare* located ?
	Ours	what **city** is *River Yare* in ?
3	Reference	who composed *Bien O Mal* ?
	Serban et al.	who is the composer of *Bien O Mal* ?
	Elsahar et al.	who is the composer of the song *Bien O Mal* ?
	Ours	who composed *Bien O Mal* ?

Figure 4: Examples of questions by different models.

frameworks on NLP tasks such as machine translation (Cho et al., 2014a) and dialog generation (Vinyals and Le, 2015). Our work is also inspired by the recent work for KBQG based on encoder-decoder frameworks. Serban et al. (2016) first proposed a neural network for mapping KB facts into natural language questions. To improve the generalization, Elsahar et al. (2018) introduced extra contexts for the input fact, which achieved significant performances. However, these contexts may make it difficult to generate questions that express the given predicate and associate with a definitive answer. Therefore, we focus on the two research issues: expressing the given predicate and referring to a definitive answer for generated questions.

Moreover, our work also borrows the idea from copy mechanisms. Point network (Vinyals et al., 2015) predicted the output sequence directly from the input, and it can not generate new words while CopyNet (Gu et al., 2016) combined copying and generating. Bao et al. (2018) proposed to copy elements in the table (KB). Elsahar et al. (2018) exploited POS copy action to better capture textual contexts. To incorporate advantages from above copy mechanisms, we introduce KB copy and context copy which can copy KB element and textual context, and they do not rely on POS tagging.

6 Conclusion and Future Work

In this paper, we focus on two crucial research issues for the task of question generation over knowledge bases: generating questions that express the given predicate and refer to definitive answers rather than alternative answers. For this purpose, we present a neural encoder-decoder model which integrates diversified off-the-shelf contexts and multi-level copy mechanisms. Moreover, we design an answer-aware loss to generate questions that refer to definitive answers. Experiments show that our model achieves state-of-the-art performance on automatic and manual evaluations.

For future work, we investigate error cases by analyzing the error distributions of 100 examples. We find that most generated questions (51%) are judged by the human to correctly express the input facts, but they unfortunately obtain low scores on the widely used metrics. It implies that it is still intractable to evaluate generated questions. Although we additionally evaluate on predicate identification and answer coverage, these metrics may be coarse and deserve further study.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61533018), the Natural Key R&D Program of China (No.2018YFC0830101), the National Natural Science Foundation of China (No.61702512) and the independent research project of National Laboratory of Pattern Recognition. This work was also supported by CCF-Tencent Open Research Fund.

References

Jun-Wei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. 2018. Table-to-text: Describing table region with natural language. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5020–5027.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder

approaches. In *SSST@EMNLP*, pages 103–111. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874. Association for Computational Linguistics.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 218–228. Association for Computational Linguistics.

Yichen Gong and Samuel Bowman. 2018. Ruminating reader: Reasoning with gated multi-hop attention. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 1–11. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 1003–1011, Stroudsburg, PA, USA. Association for Computational Linguistics.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 291–296. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

ael Petrochuk and Luke Zettlemoyer. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 554–558. Association for Computational Linguistics.

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598. Association for Computational Linguistics.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI Conference on Artificial Intelligence*.

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. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of Advances in neural information processing systems*, pages 3104–3112.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. 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 5998–6008. Curran Associates, Inc.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. Cite arxiv:1506.05869Comment: ICML Deep Learning Workshop 2015.

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