Learning to Ask Questions in Open-domain Conversational Systems with Typed Decoders

Yansen Wang[1,*], **Chenyi Liu**[1,*], **Minlie Huang**[1,†], **Liqiang Nie**[2]
[1]Conversational AI group, AI Lab., Department of Computer Science, Tsinghua University
[1]Beijing National Research Center for Information Science and Technology, China
[2]Shandong University
ys-wang15@mails.tsinghua.edu.cn;liucy15@mails.tsinghua.edu.cn;
aihuang@tsinghua.edu.cn;nieliqiang@gmail.com

1-3
3 notes:

04
Feyoe Xia

05
Feyoe Xia

Abstract

Asking good questions in large-scale, open-domain conversational systems is quite significant yet rather untouched. This task, substantially different from traditional question generation, requires to question not only with various patterns but also on diverse and relevant topics. We observe that a good question is a natural composition of *interrogatives*, *topic words*, and *ordinary words*. Interrogatives lexicalize the pattern of questioning, topic words address the key information for topic transition in dialogue, and ordinary words play syntactical and grammatical roles in making a natural sentence. We devise two typed decoders (*soft typed decoder* and *hard typed decoder*) in which a type distribution over the three types is estimated and used to modulate the final generation distribution. Extensive experiments show that the typed decoders outperform state-of-the-art baselines and can generate more meaningful questions.

1 Introduction

Learning to ask questions (or, question generation) aims to generate a question to a given input. Deciding what to ask and how is an indicator of machine understanding (Mostafazadeh et al., 2016), as demonstrated in machine comprehension (Du et al., 2017; Zhou et al., 2017b; Yuan et al., 2017) and question answering (Tang et al., 2017; Wang et al., 2017). Raising good questions is essential to conversational systems because a good system can well interact with users by asking and responding (Li et al., 2016). Furthermore, asking questions is one of the important proactive behaviors that can drive dialogues to go deeper and further (Yu et al., 2016).

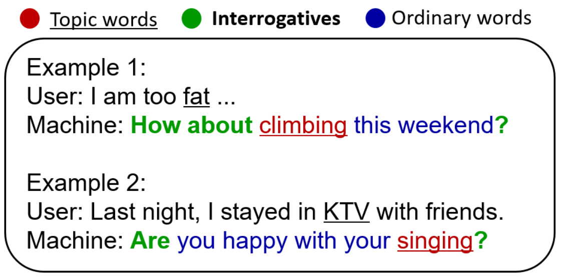
Question generation (QG) in open-domain conversational systems differs substantially from the traditional QG tasks. The ultimate goal of this task is to enhance the *interactiveness and persistence of human-machine interactions*, while for traditional QG tasks, seeking information through a generated question is the major purpose. The response to a generated question will be supplied in the following conversations, which may be novel but not necessarily occur in the input as that in traditional QG (Du et al., 2017; Yuan et al., 2017; Tang et al., 2017; Wang et al., 2017; Mostafazadeh et al., 2016). Thus, the purpose of this task is to spark novel yet related information to drive the interactions to continue.

Due to the different purposes, this task is unique in two aspects: it requires to question not only in various patterns but also about diverse yet relevant topics. **First**, there are various questioning patterns for the same input, such as Yes-no questions and Wh-questions with different interrogatives. Diversified questioning patterns make dialogue interactions richer and more flexible. Instead, traditional QG tasks can be roughly addressed by syntactic transformation (Andrenucci and Sneiders, 2005; Popowich and Winne, 2013), or implicitly modeled by neural models (Du et al., 2017). In such tasks, the information questioned on is pre-specified and usually determines the pattern of questioning. For instance, asking Who-question for a given person, or Where-question for a given location.

Second, this task requires to address much more transitional topics of a given input, which is the nature of conversational systems. For instance, for the input *"I went to dinner with my friends"*, we may question about topics such as *friend, cuisine,*

[*]Authors contributed equally to this work.
[†]Corresponding author: Minlie Huang.

price, place and *taste*. Thus, this task generally requires *scene understanding* to imagine and comprehend a scenario (e.g., *dining at a restaurant*) that can be interpreted by topics related to the input. However, in traditional QG tasks, the core information to be questioned on is pre-specified and rather static, and *paraphrasing* is more required.



● Topic words ● **Interrogatives** ● Ordinary words

Example 1:
User: I am too fat ...
Machine: **How about** climbing this weekend**?**

Example 2:
User: Last night, I stayed in KTV with friends.
Machine: **Are** you happy with your singing**?**

Figure 1: Good questions in conversational systems are a natural composition of interrogatives, topic words, and ordinary words.

Undoubtedly, asking good questions in conversational systems needs to address the above issues (*questioning with diversified patterns, and addressing transitional topics naturally in a generated question*). As shown in Figure 1, a good question is a natural composition of interrogatives, topic words, and ordinary words. Interrogatives indicate the pattern of questioning, topic words address the key information of topic transition, and ordinary words play syntactical and grammatical roles in making a natural sentence.

We thus classify the words in a question into three types: **interrogative**, **topic word**, and **ordinary word** automatically. We then devise two decoders, Soft Typed Decoder (STD) and Hard Typed Decoder (HTD), for question generation in conversational systems[1]. STD deals with word types in a latent and implicit manner, while HTD in a more explicit way. At each decoding position, we firstly estimate a type distribution over word types. STD applies a mixture of type-specific generation distributions where type probabilities are the coefficients. By contrast, HTD reshapes the type distribution by Gumbel-softmax and modulates the generation distribution by type probabilities. Our contributions are as follows:

- To the best of our knowledge, this is the first study on question generation in the setting of conversational systems. We analyze the key differences between this new task and other traditional question generation tasks.

- We devise soft and hard typed decoders to ask good questions by capturing different roles of different word types. Such typed decoders may be applicable to other generation tasks if word semantic types can be identified.

[1]To simplify the task, as a preliminary research, we consider the one-round conversational system.

2 Related Work

Traditional question generation can be seen in task-oriented dialogue system (Curto et al., 2012), sentence transformation (Vanderwende, 2008), machine comprehension (Du et al., 2017; Zhou et al., 2017b; Yuan et al., 2017; Subramanian et al., 2017), question answering (Qin, 2015; Tang et al., 2017; Wang et al., 2017; Song et al., 2017), and visual question answering (Mostafazadeh et al., 2016). In such tasks, the answer is known and is part of the input to the generated question. Meanwhile, the generation tasks are not required to predict additional topics since all the information has been provided in the input. They are applicable in scenarios such as designing questions for reading comprehension (Du et al., 2017; Zhou et al., 2017a; Yuan et al., 2017), and justifying the visual understanding by generating questions to a given image (video) (Mostafazadeh et al., 2016).

In general, traditional QG tasks can be addressed by the heuristic rule-based reordering methods (Andrenucci and Sneiders, 2005; Ali et al., 2010; Heilman and Smith, 2010), slot-filling with question templates (Popowich and Winne, 2013; Chali and Golestanirad, 2016; Labutov et al., 2015), or implicitly modeled by recent neural models(Du et al., 2017; Zhou et al., 2017b; Yuan et al., 2017; Song et al., 2017; Subramanian et al., 2017). These tasks generally do not require to generate a question with various patterns: for a given answer and a supporting text, the question type is usually decided by the input.

Question generation in large-scale, open-domain dialogue systems is relatively unexplored. Li et al. (2016) showed that asking questions in task-oriented dialogues can offer useful feedback to facilitate learning through interactions. Several questioning mechanisms were devised with hand-crafted templates, but unfortunately not applicable to open-domain conversational systems. Similar to our goal, a visual QG task is proposed to generate a question to interact with other people, given an image as input (Mostafazadeh et al., 2016).

3 Methodology

3.1 Overview

The task of question generation in conversational systems can be formalized as follows: given a user post $X = x_1 x_2 \cdots x_m$, the system should generate a natural and meaningful question $Y = y_1 y_2 \cdots y_n$ to interact with the user, formally as

$$Y^* = \underset{Y}{argmax}\, \mathcal{P}(Y|X).$$

As aforementioned, asking good questions in conversational systems requires to question with diversified patterns and address transitional topics naturally in a question. To this end, we classify the words in a sentence into three types: *interrogative*, *topic word*, and *ordinary word*, as shown in Figure 1. During training, the type of each word in a question is decided automatically[2]. We manually collected about 20 interrogatives. The verbs and nouns in a question are treated as topic words, and all the other words as ordinary words. During test, we resort to PMI (Church and Hanks, 1990) to predict a few topic words for a given post.

On top of an encoder-decoder framework, we propose two decoders to effectively use word types in question generation. The first model is *soft typed decoder (STD)*. It estimates a type distribution over word types and three type-specific generation distributions over the vocabulary, and then obtains a mixture of type-specific distributions for word generation.

The second one is a *hard* form of STD, *hard typed decoder (HTD)*, in which we can control the decoding process more explicitly by approximating the operation of *argmax* with Gumbel-softmax (Jang et al., 2016). In both decoders, the final generation probability of a word is modulated by its word type.

3.2 Encoder-Decoder Framework

Our model is based on the general encoder-decoder framework (Cho et al., 2014; Sutskever et al., 2014). Formally, the model encodes an input sequence $X = x_1 x_2 \cdots x_m$ into a sequence of hidden states $\mathbf{h}_i$, as follows,

$$\mathbf{h}_t = \mathbf{GRU}(\mathbf{h}_{t-1}, \boldsymbol{e}(x_t)),$$

[2]Though there may be errors in word type classification, we found it works well in response generation.

where GRU denotes gated recurrent units (Cho et al., 2014), and $\boldsymbol{e}(x)$ is the word vector of word x. The decoder generates a word sequence by sampling from the probability $\mathcal{P}(y_t|y_{<t}, X)$ ($y_{<t} = y_1 y_2 \cdots y_{t-1}$, the generated subsequence) which can be computed via

$$\mathcal{P}(y_t|y_{<t}, X) = \mathbf{MLP}(\mathbf{s}_t, \boldsymbol{e}(y_{t-1}), \mathbf{c}_t),$$
$$\mathbf{s}_t = \mathbf{GRU}(\mathbf{s}_{t-1}, \boldsymbol{e}(y_{t-1}), \mathbf{c}_t),$$

where $\mathbf{s}_t$ is the state of the decoder at the time step t, and this GRU has different parameters with the one of the encoder. The context vector $\mathbf{c}_t$ is an attentive read of the hidden states of the encoder as $\mathbf{c}_t = \sum_{i=1}^{T} \alpha_{t,i} \mathbf{h}_i$, where the weight $\alpha_{t,i}$ is scored by another $\mathbf{MLP}(\mathbf{s}_{t-1}, \mathbf{h}_i)$ network.

3.3 Soft Typed Decoder (STD)

In a general encoder-decoder model, the decoder tends to generate universal, meaningless questions like "*What's up?*" and "*So what?*". In order to generate more meaningful questions, we propose a soft typed decoder. It assumes that each word has a *latent type* among the set {*interrogative, topic word, ordinary word*}. The soft typed decoder firstly estimates a word type distribution over latent types in the given context, and then computes type-specific generation distributions over the entire vocabulary for different word types. The final probability of generating a word is a mixture of type-specific generation distributions where the coefficients are type probabilities.

The final generation distribution $\mathcal{P}(y_t|y_{<t}, X)$ from which a word can be sampled, is given by

$$\mathcal{P}(y_t|y_{<t}, X) = \sum_{i=1}^{k} \mathcal{P}(y_t|ty_t = c_i, y_{<t}, X) \cdot \mathcal{P}(ty_t = c_i|y_{<t}, X), \quad (1)$$

where ty_t denotes the word type at time step t and c_i is a word type. Apparently, this formulation states that the final generation probability is a mixture of the type-specific generation probabilities $\mathcal{P}(y_t|ty_t = c_i, y_{<t}, X)$, weighted by the probability of the type distribution $\mathcal{P}(ty_t = c_i|y_{<t}, X)$. We name this decoder as *soft typed decoder*. In this model, word type is latent because we do not need to specify the type of a word explicitly. In other words, each word can belong to any of the three types, but with different probabilities given the current context.

The probability distribution over word types $\mathcal{C} = \{c_1, c_2, \cdots, c_k\}$ ($k = 3$ in this paper) (termed

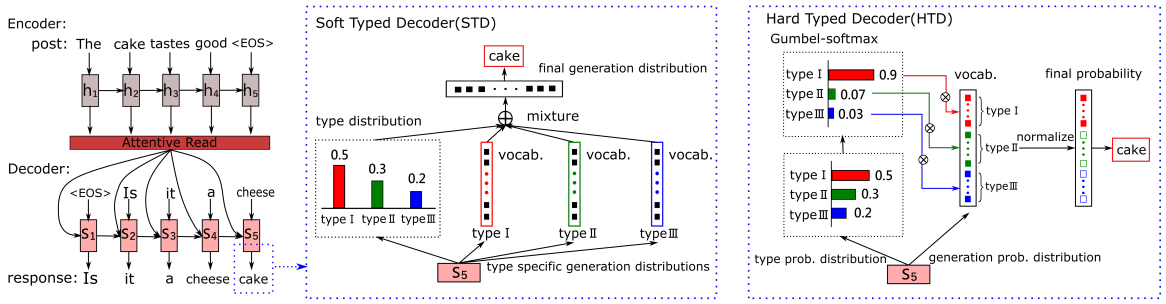


Figure 2: Illustration of STD and HTD. STD applies a mixture of type-specific generation distributions where type probabilities are the coefficients. In HTD, the type probability distribution is reshaped by Gumbel-softmax and then used to modulate the generation distribution. In STD, the generation distribution is over the same vocabulary whereas dynamic vocabularies are applied in HTD.

as *type distribution*) is given by

$$\mathcal{P}(ty_t|y_{<t}, X) = softmax(\mathbf{W}_0\mathbf{s}_t + \mathbf{b}_0), \quad (2)$$

where s_t is the hidden state of the decoder at time step t, $\mathbf{W}_0 \in R^{k\times d}$, and d is the dimension of the hidden state.

The type-specific generation distribution is given by

$$\mathcal{P}(y_t|ty_t = c_i, y_{<t}, X) = softmax(\mathbf{W}_{c_i}\mathbf{s}_t + \mathbf{b}_{c_i}),$$

where $\mathbf{W}_{c_i} \in R^{|V|\times d}$ and $|V|$ is the size of the entire vocabulary. Note that the type-specific generation distribution is parameterized by $\mathbf{W}_{c_i}$, indicating that the distribution for each word type has its own parameters.

Instead of using a single distribution $\mathcal{P}(y_t|y_{<t}, X)$ as in a general Seq2Seq decoder, our soft typed decoder enriches the model by applying multiple type-specific generation distributions. This enables the model to express more information about the next word to be generated. Also note that the generation distribution is over the same vocabulary, and therefore there is no need to specify word types explicitly.

3.4 Hard Typed Decoder (HTD)

In the soft typed decoder, we assume that each word is a distribution over the word types. In this sense, the type of a word is **implicit**. We do not need to specify the type of each word **explicitly**. In the hard typed decoder, words in the entire vocabulary are dynamically classified into three types for each post, and the decoder first estimates a type distribution at each position and then generates a word with the highest type probability. This process can be formulated as follows:

$$c^* = \arg\max_{c_i} \mathcal{P}(ty_t = c_i|y_{<t}, X), \quad (3)$$

$$\mathcal{P}(y_t|y_{<t}, X) = \mathcal{P}(y_t|ty_t = c^*, y_{<t}, X). \quad (4)$$

This is essentially the *hard* form of Eq. 1, which just selects the type with the maximal probability. However, this *argmax* process may cause two problems. First, such a cascaded decision process (firstly selecting the most probable word type and secondly choosing a word from that type) may lead to severe grammatical errors if the first selection is wrong. Second, *argmax* is discrete and non-differentiable, and it breaks the back-propagation path during training.

To make best use of word types in *hard typed decoder*, we address the above issues by applying *Gumbel-Softmax* (Jang et al., 2016) to approximate the operation of *argmax*. There are several steps in the decoder (see Figure 2):

First, the type of each word (*interrogative, topic, or ordinary*) in a question is decided automatically during training, as aforementioned.

Second, the generation probability distribution is estimated as usual,

$$\mathcal{P}(y_t|y_{<t}, X) = softmax(\mathbf{W}_0\mathbf{s}_t + \mathbf{b}_0). \quad (5)$$

Further, the type probability distribution at each decoding position is estimated as follows,

$$\mathcal{P}(ty_t|y_{<t}, X) = softmax(\mathbf{W}_1\mathbf{s}_t + \mathbf{b}_1). \quad (6)$$

Third, the generation probability for each word is modulated by its corresponding type probability:

$$\mathcal{P}'(y_t|y_{<t}, X) = \mathcal{P}(y_t|y_{<t}, X)\cdot \boldsymbol{m}(y_t),$$
$$\boldsymbol{m}(y_t) = \begin{cases} 1 & , c(y_t) = c^* \\ 0 & , c(y_t) \neq c^* \end{cases} \quad (7)$$

where $c(y_t)$ looks up the word type of word y_t, and c^* is the type with the highest probability as defined in Eq. 3. This formulation has exactly the effect of *argmax*, where the decoder will only generate words of type with the highest probability.

To make $\mathcal{P}^*(y_t|y_{<t}, X)$ a distribution, we normalize these values by a normalization factor Z:

$$Z = \frac{1}{\sum_{y_t \in \mathcal{V}} \mathcal{P}'(y_t|y_{<t}, X)}$$

where $\mathcal{V}$ is the decoding vocabulary. Then, the final probability can be denoted by

$$\mathcal{P}^*(y_t|y_{<t}, X) = Z \cdot \mathcal{P}'(y_t|y_{<t}, X). \quad (8)$$

As mentioned, in order to have an effect of *argmax* but still maintain the differentiability, we resort to *Gumbel-Softmax* (Jang et al., 2016), which is a differentiable surrogate to the *argmax* function. The type probability distribution is then adjusted to the following form:

$$\boldsymbol{m}(y_t) = \mathbf{GS}(\mathcal{P}(ty_t = c(y_t)|y_{<t}, X)),$$
$$\mathbf{GS}(\pi_i) = \frac{e^{(log(\pi_i)+g_i)/\tau}}{\sum_{j=1}^{k} e^{(log(\pi_j)+g_j)/\tau}}, \quad (9)$$

where $\pi_1, \pi_2, \cdots, \pi_k$ represents the probabilities of the original categorical distribution, g_j are i.i.d samples drawn from Gumbel(0,1)[3] and τ is a constant that controls the smoothness of the distribution. When $\tau \to 0$, Gumbel-Softmax performs like argmax, while if $\tau \to \infty$, Gumbel-Softmax performs like a uniform distribution. In our experiments, we set τ a constant between 0 and 1, making Gumbel-Softmax smoother than argmax, but sharper than normal softmax.

Note that in HTD, we apply dynamic vocabularies for different responses during training. The words in a response are classified into the three types dynamically. A specific type probability will only affect the words of that type. During test, for each post, topic words are predicted with PMI, interrogatives are picked from a small dictionary, and the rest of words in the vocabulary are treated as ordinary words.

[3]If $u \sim Uniform(0, 1)$, then $g = -log(-log(u)) \sim Gumbel(0, 1)$.

3.5 Loss Function

We adopt negative data likelihood (equivalent to cross entropy) as the loss function, and additionally, we apply supervision on the mixture weights of word types, formally as follows:

$$\Phi_1 = \sum_t -\log \mathcal{P}(y_t = \tilde{y}_t|y_{<t}, X), \quad (10)$$
$$\Phi_2 = \sum_t -\log \mathcal{P}(ty_t = \widetilde{ty}_t|y_{<t}, X), \quad (11)$$
$$\Phi = \Phi_1 + \lambda\Phi_2, \quad (12)$$

where $\widetilde{ty}_t$ represents the reference word type and $\tilde{y}_t$ represents the reference word at time t. λ is a factor to balance the two loss terms, and we set λ=0.8 in our experiments.

Note that for HTD, we substitute $\mathcal{P}^*(y_t = w_j|y_{<t}, X)$ (as defined by Eq. 8) into Eq. 10.

3.6 Topic Word Prediction

The only difference between training and inference is the means of choosing topic words. During training, we identify the nouns and verbs in a response as topic words; whereas during inference, we adopt PMI (Church and Hanks, 1990) and $Rel(k_i, X)$ to predict a set of topic words k_i for an input post X, as defined below:

$$PMI(w_x, w_y) = log\frac{p(w_x, w_y)}{p_1(w_x) * p_2(w_y)},$$
$$Rel(k_i, X) = \sum_{w_x \in X} e^{PMI(w_x, k_i)},$$

where $p_1(w)/p_2(w)$ represent the probability of word w occurring in a post/response, respectively, and $p(w_x, w_y)$ is the probability of word w_x occurring in a post and w_y in a response.

During inference, we predict at most 20 topic words for an input post. Too few words will affect the grammaticality since the predicted set contains infrequent topic words, while too many words introduce more common topics leading to more general responses.

4 Experiment

4.1 Dataset

To estimate the probabilities in PMI, we collected about 9 million post-response pairs from Weibo. To train our question generation models, we distilled the pairs whereby the responses are in question form with the help of around 20 hand-crafted templates. The templates contain a list of interrogatives and other implicit questioning patterns. Such patterns detect sentences led by words like *what, how many, how about* or sentences ended with a *question mark*. After that, we removed the pairs whose responses are universal questions that can be used to reply many different posts. This is a simple yet effective way to avoid situations where the type probability distribution is dominated by interrogatives and ordinary words.

Ultimately, we obtained the dataset comprising about 491,000 post-response pairs. We randomly selected 5,000 pairs for testing and another 5,000 for validation. The average number of words in post/response is 8.3/9.3 respectively. The dataset contains 66,547 different words, and 18,717 words appear more than 10 times. The dataset is available at: `http://coai.cs.tsinghua.edu.cn/hml/dataset/`.

4.2 Baselines

We compared the proposed decoders with four state-of-the-art baselines.

Seq2Seq: A simple encoder-decoder with attention mechanisms (Luong et al., 2015).

MA: The mechanism-aware (MA) model applies multiple responding mechanisms represented by real-valued vectors (Zhou et al., 2017a). The number of mechanisms is set to 4 and we randomly picked one response from the generated responses for evaluation to avoid selection bias.

TA: The topic-aware (TA) model generates informative responses by incorporating topic words predicted from the input post (Xing et al., 2017).

ERM: Elastic responding machine (ERM) adaptively selects a subset of responding mechanisms using reinforcement learning (Zhou et al., 2018a). The settings are the same as the original paper.

4.3 Experiment Settings

Parameters were set as follows: we set the vocabulary size to $20,000$ and the dimension of word vectors as 100. The word vectors were pretrained with around 9 million post-response pairs from Weibo and were being updated during the training of the decoders. We applied the 4-layer GRU units (hidden states have 512 dimensions). These settings were also applied to all the baselines. λ in Eq. 12 is 0.8. We set different values of τ in Gumbel-softmax at different stages of training. At the early stage, we set τ to a small value (0.6) to obtain a sharper reformed distribution (more like argmax). After several steps, we set τ to a larger value (0.8) to apply a more smoothing distribution. Our codes are available at: `https://github.com/victorywys/Learning2Ask_TypedDecoder`.

4.4 Automatic Evaluation

We conducted automatic evaluation over the $5,000$ test posts. For each post, we obtained responses from the six models, and there are $30,000$ post-response pairs in total.

4.4.1 Evaluation Metrics

We adopted *perplexity* to quantify how well a model fits the data. Smaller values indicate better performance. To evaluate the diversity of the responses, we employed *distinct-1* and *distinct-2* (Li et al., 2015). These two metrics calculates the proportion of the total number of distinct unigrams or bigrams to the total number of generated tokens in all the generated responses.

Further, we calculated the proportion of the responses containing at least one topic word in the list predicted by PMI. This is to evaluate the ability of addressing topic words in response. We term this metric as *topical response ratio (TRR)*. We predicted 20 topic words with PMI for each post.

4.4.2 Results

Comparative results are presented in Table 1. STD and HTD perform fairly well with lower perplexities, higher distinct-1 and distinct-2 scores, and remarkably better topical response ratio (TRR). Note that MA has the lowest perplexity because the model tends to generate more universal responses.

Model	Perplexity	Distinct-1	Distinct-2	TRR
Seq2Seq	63.71	0.0573	0.0836	6.6%
MA	**54.26**	0.0576	0.0644	4.5%
TA	58.89	0.1292	0.1781	8.7%
ERM	67.62	0.0355	0.0710	4.5%
STD	56.77	0.1325	0.2509	12.1%
HTD	56.10	**0.1875**	**0.3576**	**43.6%**

Table 1: Results of automatic evaluation.

Our decoders have better distinct-1 and distinct-2 scores than baselines do, and HTD performs much better than the strongest baseline TA. Noticeably, the means of using topic information in our models differs substantially from that in TA. Our decoders predict whether a topic word should be decoded at each position, whereas TA takes as

Models	Appropriateness			Richness			Willingness		
	Win (%)	Lose (%)	Tie (%)	Win (%)	Lose (%)	Tie (%)	Win (%)	Lose (%)	Tie (%)
STD vs. Seq2Seq	42.0	38.6	19.4	37.2**	15.2	47.6	45.4*	38.6	16.0
STD vs. MA	39.6*	31.2	29.2	32.6**	16.8	50.6	49.4**	27.0	23.6
STD vs. TA	42.2	40.0	17.8	49.0**	5.4	45.6	47.6*	40.2	12.2
STD vs. ERM	43.4*	34.4	22.2	60.6**	13.2	26.2	43.2*	36.8	20.0
HTD vs. Seq2Seq	50.6**	30.6	18.8	46.0**	10.2	43.8	58.4**	33.2	8.4
HTD vs. MA	54.8**	24.4	20.8	45.0**	17.0	38.0	67.0**	18.0	15.0
HTD vs. TA	52.0**	38.2	9.8	55.0**	5.4	39.6	62.6**	31.0	6.4
HTD vs. ERM	64.8**	23.2	12.0	72.2**	8.4	19.4	56.6**	36.6	6.8
HTD vs. STD	52.0**	33.0	15.0	38.0**	26.2	35.8	61.8**	30.6	7.6

Table 2: Annotation results. Win for "A vs. B" means A is better than B. Significance tests with Z-test were conducted. Values marked with * means *p-value* < 0.05, and ** for *p-value* < 0.01.

input topic word embeddings at all decoding positions.

Our decoders have remarkably better topic response ratios (TRR), indicating that they are more likely to include topic words in generation.

4.5 Manual Evaluation

We resorted to a crowdsourcing service for manual annotation. 500 posts were sampled for manual annotation[4]. We conducted pair-wise comparison between two responses generated by two models for the same post. In total, there are 4,500 pairs to be compared. For each response pair, five judges were hired to give a preference between the two responses, in terms of the following three metrics. Tie was allowed, and system identifiers were masked during annotation.

4.5.1 Evaluation Metrics

Each of the following metrics is evaluated independently on each pair-wise comparison:

Appropriateness: measures whether a question is reasonable in logic and content, and whether it is questioning on the key information. Inappropriate questions are either irrelevant to the post, or have grammatical errors, or universal questions.

Richness: measures whether a response contains topic words that are relevant to a given post.

Willingness to respond: measures whether a user will respond to a generated question. This metric is to justify how likely the generated questions can elicit further interactions. If people are willing to respond, the interactions can go further.

[4]During the sampling process, we removed those posts that are only interpretable with other context or background.

4.5.2 Results

The label of each pair-wise comparison is decided by majority voting from five annotators. Results shown in Table 2 indicate that STD and HTD outperform all the baselines in terms of all the metrics. This demonstrates that our decoders produce more appropriate questions, with richer topics. Particularly, our decoders have substantially better *willingness scores*, indicating that questions generated by our models are more likely to elicit further interactions. Noticeably, HTD outperforms STD significantly, indicating that it is beneficial to specify word types explicitly and apply dynamic vocabularies in generation.

We also observed that STD outperforms Seq2Seq and TA, but the differences are not significant in appropriateness. This is because STD generated about 7% non-question responses which were judged as inappropriate, while Seq2Seq and TA generated universal questions (inappropriate too but beat STD in annotation) to these posts.

4.5.3 Annotation Statistics

The proportion of the pair-wise annotations in which at least three of five annotators assign the same label to a record is 90.57%/93.11%/96.62% for appropriateness/ richness/willingness, respectively. The values show that we have fairly good agreements with majority voting.

4.6 Questioning Pattern Distribution

To analyze whether the model can question with various patterns, we manually annotated the questioning patterns of the responses to 100 sampled posts. The patterns are classified into 11 types including Yes-No, How-, Why-, What-, When-, and Who- questions. We then calculated the KL divergence between the pattern type distribution by a model and that by human (i.e., gold responses).

Results in Table 3 show that the pattern distribution by our model is closer to that in human-written responses, indicating that our decoders can better learn questioning patterns from human language. Further investigation reveals that the baselines tend to generate simple questions like *What?*(什么？) or *Really?*(真的吗), and constantly focus on using one or two question patterns whereas our decoders use more diversified patterns as appeared in the human language.

Model	Seq2Seq	TA	MA	ERM	STD	HTD
KLD	0.431	0.416	0.332	0.276	**0.072**	0.140

Table 3: KL divergence between the questioning pattern distribution by a model and that by human.

4.7 Examples of the Generated Questions

Table 4 presents some generated questions by our decoders, which are more appropriate. On the contrary, Seq2Seq, MA and ERM tend to generate more universal questions. These examples also clearly show that asking questions in open-domain conversational systems requires *scene understanding*, which is verified by this scene example of *singing at karaoke*(在卡拉ok唱歌).

Post:	我们昨天去唱歌了 We went to karaoke yesterday.
Seq2Seq:	**什么时候?** **When?**
MA:	你**怎么**知道我是在问 **How** did you know I am questioning you?
TA:	**什么?** **What?**
ERM:	**什么**情况**?** **What** happened **?**
STD:	去**哪里**唱歌**?** **Where** did you sing karaoke**?**
HTD:	你们**几个**人唱歌**?** **How many** people were singing with you**?**

Table 4: Generated question examples. Interrogative words are bolded and topic words underlined.

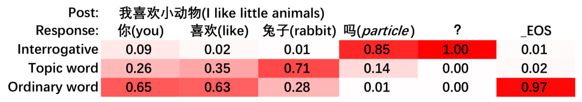
Post-1:	不好不好。哼！讨厌你啦。 No, no. Humph, I hate you!
Human:	我**怎么**了**?** **What** is wrong with me**?**
STD:	我**怎么**了**?** **What** is wrong with me**?**
HTD:	我**怎么**了**?** **What** is wrong with me**?**
Post-2:	刚用了杀虫剂，现在已经有效果了。 I just sprayed the insecticide, and now the effect appears.
Human:	这么快!效果**怎么样?** So quick! **What** is the effect**?**
STD:	效果**如何?** **What** is the effect**?**
HTD:	效果**如何?** **What** is the effect**?**
Post-3:	我在体育中心上班。 I work at the sports center.
Human:	你每天从家过去要**多久?** **How** long does it take you to go there from home**?**
STD:	你是**什么**部门的啊**?** **Which** department are you from**?**
HTD:	那你是体育学院的学生**吗?** **Are** you a student from the College of Physical Education**?**
Post-4:	今天的寿司真好吃。 The sushi today is so delicious!
Human:	我记得你不喜欢吃日式料理的不是**吗?** I remember you didn't like eating Japanese food, **did** you**?**
STD:	你**什么**时候来无锡，我请你。 **When** will you come to Wuxi**?** It is my treat.
HTD:	是**吗?**改天尝尝! Really**?** I will try sometime!

Table 5: Examples for typical questioning patterns. Interrogative words in response are bolded and topic words are underlined.

To further understand how to ask good questions in conversational systems, we presented more generated question examples by human and our typed decoders. The first example (Post-1) in Table 5 shows that a good question can be composed of only interrogatives and ordinary words. This shows the complexity of this task and also the necessity of modeling interrogatives. Post-2 indicates a typical questioning pattern which is questioning on a particular topic word (效果-*effect*) of the input. While for Post-3, the questions are asking about transitional topics of the input (上班-*work* → 部门-*department;* 体育中心-*sports center* → 体育学院-*college of Physical Education*), indicating a typical case of **topic transition** in our task (also seen in Post-4, 寿司-sushi →日式料理-Japanese food). This example also demonstrates that for the same input, there are **various questioning patterns**: a How-question asked by human, a Which-question by STD, and a Yes-No question by HTD. As for Post-4, the gold question requires a background that is only shared between the poster and responder, while STD and HTD tend to raise more general questions due to the lack of such shared knowledge.

4.8 Visualization of Type Distribution

To gain more insights into how a word type influence the generation process, we visualized the type probability at each decoding position in HTD. This example (Figure 3) shows that the model can capture word types well at different positions. For instance, at the first and second positions, ordinary words have the highest probabilities for generating 你-*you* and 喜欢-*like*, and at the third position, a topic word 兔子-*rabbit* is predicted while the last two positions are for interrogatives (a particle and a question mark).



Post:	我喜欢小动物(I like little animals)					
Response:	你(you)	喜欢(like)	兔子(rabbit)	吗(*particle*)	?	_EOS
Interrogative	0.09	0.02	0.01	0.85	1.00	0.01
Topic word	0.26	0.35	0.71	0.14	0.00	0.02
Ordinary word	0.65	0.63	0.28	0.01	0.00	0.97

Figure 3: Type distribution examples from HTD. The generated question is "你喜欢兔子吗？ *do you like rabbit?*". *_EOS* means end of sentence.

4.9 Error Analysis

We presented error type distribution by manually analyzing 100 bad responses sampled from STD and HTD respectively, where *bad* means the response by our model is worse than that by some baseline during the pair-wise annotation.

There are 4 typical error types: *no topic words (NoT)* in a response (mainly universal questions), *wrong topics (WrT)* where topic words are irrelevant, *type generation error (TGE)* where a wrong word type is predicted (See Eq. 2) and it causes grammatical errors, and *other errors*.

Error Type	NoT	WrT	TGE	Others
STD	34%	34%	29%	3%
HTD	29%	39%	29%	3%

Table 6: Error type distribution.

The error distribution is shown in Table 6. For STD, most of the errors are attributed to no topic or wrong topics, while for HTD, the majority of errors fall into *wrong topics*.

NoT	**Post-1:**	今天好开心啊! I am so happy today!
	STD:	你**怎么**知道**?** **How** do you know **?**
WrT	**Post-2:**	海报非常棒，期待若曦与我们男人的首度合作。 The poster is great and we look forward to our first cooperation with Ruoxi.
	HTD:	你海报**怎么**样啊**?** **How** about your poster **?**
TGE	**Post-3:**	又生病啦?吃点药就好了。 Got sick again? Just take some medicine and you'll be fine soon.
	STD:	我也不知道**怎么**回事。 I don't know **what** happened.
	HTD:	肠胃痛了，上火**吗?** Stomach aching, ulcer **?**

Table 7: Cases for the error types with interrogative words bolded and topic words underlined.

There are typical cases for these error types: (1) Posts such as *"I am so happy today!"* contains no topic words or rare topic words. In this case, our method is unable to predict the topic words so that the models tend to generate universal questions. This happens more frequently in STD because the topic words are not specified explicitly. (2) Posts contains multiple topic words, but the model sometimes focuses on an inappropriate one. For instance, for Post-2 in Table 7, HTD focused on 海报-*poster* but 合作-*cooperation* is a proper one to be focused on. (3) For complex posts, the models failed to predict the correct word type in response. For Post-3, STD generated a declarative sentence and HTD generated a question which, however, is not adequate within the context.

These cases show that controlling the questioning patterns and the informativeness of the content faces with the compatibility issue, which is challenging in language generation. These errors are also partially due to the imperfect ability of topic word prediction by PMI, which is challenging itself in open-domain conversational systems.

5 Conclusion and Future Work

We present two typed decoders to generate questions in open-domain conversational systems. The decoders firstly estimate a type distribution over word types, and then use the type distribution to modulate the final word generation distribution. Through modeling the word types in language generation, the proposed decoders are able to question with various patterns and address novel yet related transitional topics in a generated question. Results show that our models can generate more appropriate questions, with richer topics, thereby more likely to elicit further interactions.

The work can be extended to multi-turn conversation generation by including an additional detector predicting when to ask a question. The detector can be implemented by a classifier or some heuristics. Furthermore, the typed decoders are applicable to the settings where word types can be easily obtained, such as in emotional text generation (Ghosh et al., 2017; Zhou et al., 2018b).

Acknowledgements

This work was partly supported by the National Science Foundation of China under grant No.61272227/61332007 and the National Basic Research Program (973 Program) under grant No. 2013CB329403. We would like to thank Prof. Xiaoyan Zhu for her generous support.

References

Husam Ali, Yllias Chali, and Sadid A Hasan. 2010. Automation of question generation from sentences. In *Proceedings of QG2010: The Third Workshop on Question Generation*. pages 58–67.

A. Andrenucci and E. Sneiders. 2005. Automated question answering: review of the main approaches. In *ICITA*. pages 514–519.

Yllias Chali and Sina Golestanirad. 2016. Ranking automatically generated questions using common human queries. In *INLG*. pages 217–221.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*. pages 1724–1734.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics* 16(1):22–29.

Sérgio Curto, Ana Cristina Mendes, and Luísa Coheur. 2012. Question generation based on lexico-syntactic patterns learned from the web. *Dialogue Discourse* .

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

Sayan Ghosh, Mathieu Chollet, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. 2017. Affect-lm: A neural language model for customizable affective text generation. In *ACL*. pages 634–642.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *NAACL HLT*. pages 609–617.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* .

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

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*. pages 110–119.

Jiwei Li, Alexander H Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. 2016. Learning through dialogue interactions by asking questions. *arXiv preprint arXiv:1612.04936* .

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*. pages 1412–1421.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *ACL*. pages 1802–1813.

David Lindberg Fred Popowich and John Nesbit Phil Winne. 2013. Generating natural language questions to support learning on-line. *ENLG* pages 105–114.

Haocheng Qin. 2015. *Question Paraphrase Generation for Question Answering System*. Master's thesis, University of Waterloo.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058* .

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2017. Neural models for key phrase detection and question generation. *arXiv preprint arXiv:1706.04560* .

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. pages 3104–3112.

Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* .

Lucy Vanderwende. 2008. The importance of being important: Question generation. In *Proceedings of the 1st Workshop on the Question Generation Shared Task Evaluation Challenge*.

Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450* .

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *AAAI*. pages 3351–3357.

Zhou Yu, Ziyu Xu, Alan W Black, and Alex I. Rudnicky. 2016. Strategy and policy learning for non-task-oriented conversational systems. In *SIGDIAL*. pages 404–412.

Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. In *The Workshop on Representation Learning for NLP*. pages 15–25.

Ganbin Zhou, Ping Luo, Rongyu Cao, Fen Lin, Bo Chen, and Qing He. 2017a. Mechanism-aware neural machine for dialogue response generation. In *AAAI*. pages 3400–3407.

Ganbin Zhou, Ping Luo, Yijun Xiao, Fen Lin, Bo Chen, and Qing He. 2018a. Elastic responding machine for dialog generation with dynamically mechanism selecting. In *AAAI Conference on Artificial Intelligence, AAAI*.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018b. Emotional chatting machine: Emotional conversation generation with internal and external memory. *AAAI* .

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017b. Neural question generation from text: A preliminary study. *arXiv preprint arXiv:1704.01792* .

Learning to Ask Questions in Open-domain Conversational Systems with Typed Decoders

Yansen Wang1,⋆, Chenyi Liu1,⋆, Minlie Huang1,†, Liqiang Nie2

01 Feyoe Xia Page 1

1/8/2019 12:53

02 Feyoe Xia Page 1

1/8/2019 12:53

03 Feyoe Xia Page 1

1/8/2019 12:53

04 Feyoe Xia Page 1

1/8/2019 12:53

05 Feyoe Xia Page 1

1/8/2019 12:53

06 Feyoe Xia Page 3

14/7/2019 2:07

07 Feyoe Xia Page 3

1/8/2019 12:53

08 Feyoe Xia Page 3

14/7/2019 2:10

09 Feyoe Xia Page 3

14/7/2019 2:12

10 Feyoe Xia Page 3

14/7/2019 2:21

11 Feyoe Xia

14/7/2019 3:29