Generating Informative Responses with Controlled Sentence Function

Pei Ke[1], **Jian Guan**[2], **Minlie Huang**[1]*, **Xiaoyan Zhu**[1]
[1]Conversational AI group, AI Lab., Dept. of Computer Science, Tsinghua University
[1]Beijing National Research Center for Information Science and Technology, China
[2]Dept. of Physics, Tsinghua University, Beijing 100084, China
kepei1106@outlook.com, guanj15@mails.tsinghua.edu.cn
aihuang@tsinghua.edu.cn, zxy-dcs@tsinghua.edu.cn

Abstract

Sentence function is a significant factor to achieve the purpose of the speaker, which, however, has not been touched in large-scale conversation generation so far. In this paper, we present a model to generate informative responses with controlled sentence function. Our model utilizes a continuous latent variable to capture various word patterns that realize the expected sentence function, and introduces a type controller to deal with the compatibility of controlling sentence function and generating informative content. Conditioned on the latent variable, the type controller determines the type (i.e., *function-related*, *topic*, and *ordinary* word) of a word to be generated at each decoding position. Experiments show that our model outperforms state-of-the-art baselines, and it has the ability to generate responses with both controlled sentence function and informative content.

1 Introduction

Sentence function is an important linguistic feature and a typical taxonomy in terms of the purpose of the speaker (Rozakis, 2003). There are four major function types in the language including *interrogative, declarative, imperative*, and *exclamatory*, as described in (Rozakis, 2003). Each sentence function possesses its own structure, and transformation between sentence functions needs a series of changes in word order, syntactic patterns and other aspects (Akmajian, 1984; Yule, 2010).

Since sentence function is regarding the purpose of the speaker, it can be a significant factor indicating the conversational purpose during interactions, but surprisingly, this problem is rather untouched in dialogue systems. As shown in Figure 1, responses with different functions can be used to achieve different conversational purposes: **Interrogative** responses can be used to acquire further information from the user; **imperative** responses are used to make requests, directions, instructions or invitations to elicit further interactions; and **declarative** responses commonly make statements to state or explain something.[1] Interrogative and imperative responses can be used to avoid stalemates (Li et al., 2016b), which can be viewed as important proactive behaviors in conversation (Yu et al., 2016). Thus, conversational systems equipped with the ability to control the sentence function can adjust its strategy for different purposes within different contexts, behave more proactively, and may lead the dialogue to go further.

Post		I'm really hungry now.
Response	**Interrogative**	**What did** you have at breakfast**?**
	Imperative	**Let's** have dinner together**!**
	Declarative	Me, **too. But** you ate too much at lunch.

Figure 1: Responses with three sentence functions. Function-related words are in **red**, topic words in blue, and others are ordinary words.

Generating responses with controlled sentence functions differs significantly from other tasks on controllable text generation (Hu et al., 2017; Ficler and Goldberg, 2017; Asghar et al., 2017; Ghosh et al., 2017; Zhou and Wang, 2017; Dong et al., 2017; Murakami et al., 2017). These studies, involving the control of sentiment polarity, emotion, or tense, fall into **local** control, more or less, because the controllable variable can be *locally* re-

*Corresponding author: Minlie Huang.

[1] Note that we did not include the *exclamatory* category in this paper because an exclamatory sentence in conversation is only a strong emotional expression of the original sentence with few changes.

flected by decoding local variable-related words, e.g., *terrible* for negative sentiment (Hu et al., 2017; Ghosh et al., 2017), *glad* for happy emotion (Zhou et al., 2018; Zhou and Wang, 2017), and *was* for past tense (Hu et al., 2017). By contrast, sentence function is a **global** attribute of text, and controlling sentence function is more challenging in that it requires to adjust the **global structure** of the entire text, including changing word order and word patterns.

Controlling sentence function in conversational systems faces another challenge: in order to generate informative and meaningful responses, it has to deal with the compatibility of the sentence function and the content. Similar to most existing neural conversation models (Li et al., 2016a; Mou et al., 2016; Xing et al., 2017), we are also struggling with universal and meaningless responses for different sentence functions, e.g., "*Is that right?*" for interrogative responses, "*Please!*" for imperative responses and "*Me, too.*" for declarative responses. The lack of meaningful topics in responses will definitely degrade the utility of the sentence function so that the desired conversational purpose can not be achieved. Thus, the task needs to generate responses with both informative content and controllable sentence functions.

In this paper, we propose a conversation generation model to deal with the global control of sentence function and the compatibility of controlling sentence function and generating informative content. We devise an encoder-decoder structure equipped with a latent variable in conditional variational autoencoder (CVAE) (Sohn et al., 2015), which can not only project different sentence functions into different regions in a latent space, but also capture various word patterns within each sentence function. The latent variable, supervised by a discriminator with the expected function label, is also used to realize the global control of sentence function. To address the compatibility issue, we use a type controller which lexicalizes the sentence function and the content explicitly. The type controller estimates a distribution over three word types, i.e., **function-related**, **topic**, and **ordinary** words. During decoding, the word type distribution will be used to modulate the generation distribution in the decoder. The type sequence of a response can be viewed as an abstract representation of sentence function. By this means, the model has an explicit and strong control on the function and the content. Our contributions are as follows:

- We investigate how to control sentence functions to achieve different conversational purposes in open-domain dialogue systems. We analyze the difference between this task and other controllable generation tasks.
- We devise a structure equipped with a latent variable and a type controller to achieve the global control of sentence function and deal with the compatibility of controllable sentence function and informative content in generation. Experiments show the effectiveness of the model.

2 Related Work

Recently, language generation in conversational systems has been widely studied with sequence-to-sequence (seq2seq) learning (Sutskever et al., 2014; Bahdanau et al., 2015; Vinyals and Le, 2015; Shang et al., 2015; Serban et al., 2016, 2017). A variety of methods has been proposed to address the important issue of content quality, including enhancing diversity (Li et al., 2016a; Zhou et al., 2017) and informativeness (Mou et al., 2016; Xing et al., 2017) of the generated responses.

In addition to the content quality, controllability is a critical problem in text generation. Various methods have been used to generate texts with controllable variables such as sentiment polarity, emotion, or tense (Hu et al., 2017; Ghosh et al., 2017; Zhou and Wang, 2017; Zhou et al., 2018) . There are mainly two solutions to deal with controllable text generation. First, the variables to be controlled are embedded into vectors which are then fed into the models to reflect the characteristics of the variables (Ghosh et al., 2017; Zhou et al., 2018). Second, latent variables are used to capture the information of controllable attributes as in the variational autoencoders (VAE) (Zhou and Wang, 2017). (Hu et al., 2017) combined the two techniques by disentangling a latent variable into a categorical code and a random part to better control the attributes of the generated text.

The task in this paper differs from the above tasks in two aspects: (1) Unlike other tasks that realize controllable text generation by decoding attribute-related words locally, our task requires to not only decode function-related words, but also

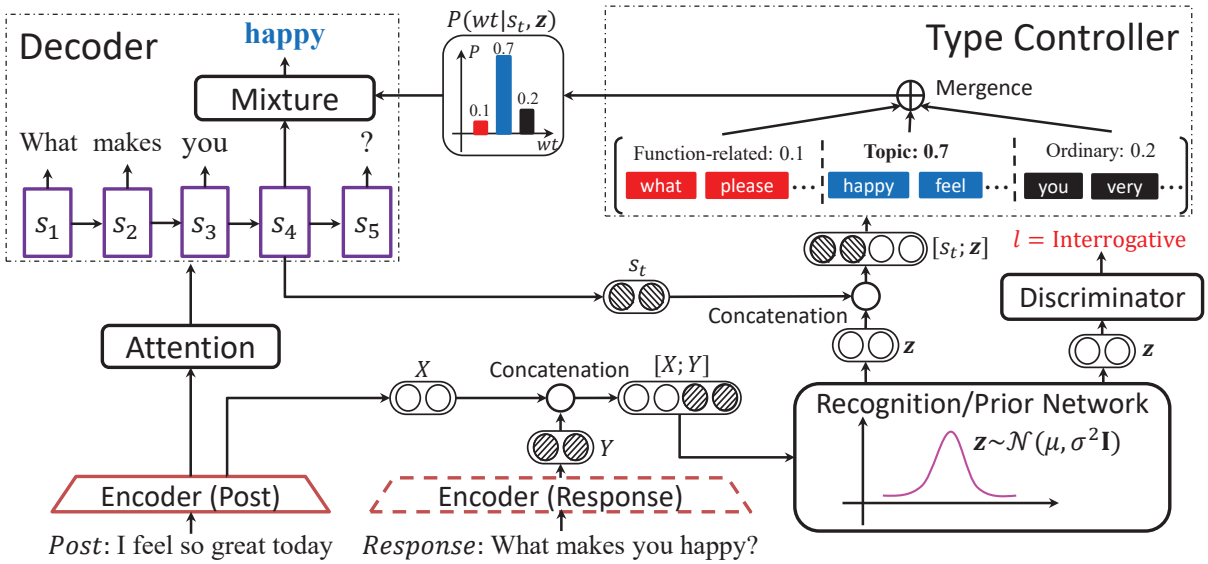

Figure 2: Model overview. During training, the latent variable z is sampled from the recognition network which is supervised by the function label in the discriminator. In the type controller, the latent variable and the decoder's state are used to estimate a type distribution which modulates the final generation distribution. During test, z is sampled from the prior network whose input is only the post. The response encoder in the dotted box appears only in training.

plan the words globally to realize the function type to be controlled. (2) The compatibility of controllable variables and content quality is less studied in the literature. The most similar work in (Zhao et al., 2017) proposed to control the dialogue act of a response, which is also a global attribute. However, the model controls dialog act by directly feeding a latent variable into the decoder, instead, our model has a stronger control on the generation process via a type controller in which words of different types are concretely modeled.

3 Model

3.1 Task Definition and Model Overview

Our problem is formulated as follows: given a post $X = x_1 x_2 \cdots x_n$ and a sentence function category l, our task is to generate a response $Y = y_1 y_2 \cdots y_m$ that is not only coherent with the specified function category l but also informative in content. We denote $\boldsymbol{c}$ as the concatenation of all the input information, i.e. $\boldsymbol{c} = [X; l]$. Essentially, the goal is to estimate the conditional probability:

$$P(Y, \boldsymbol{z}|\boldsymbol{c}) = P(\boldsymbol{z}|\boldsymbol{c}) \cdot P(Y|\boldsymbol{z}, \boldsymbol{c}) \quad (1)$$

The latent variable $\boldsymbol{z}$ is used to capture the sentence function of a response. $P(\boldsymbol{z}|\boldsymbol{c})$, parameterized as the prior network in our model, indicates the sampling process of $\boldsymbol{z}$, i.e., drawing $\boldsymbol{z}$ from $P(\boldsymbol{z}|\boldsymbol{c})$. And $P(Y|\boldsymbol{z}, \boldsymbol{c}) = \prod_{t=1}^{m} P(y_t|y_{<t}, \boldsymbol{z}, \boldsymbol{c})$ is applied to model the generation of the response Y conditioned on the latent variable $\boldsymbol{z}$ and the input $\boldsymbol{c}$, which is implemented by a decoder in our model.

Figure 2 shows the overview of our model. As aforementioned, the model is constructed in the encoder-decoder framework. The encoder takes a post and a response as input, and obtains the hidden representations of the input. The recognition network and the prior network, adopted from the CVAE framework (Sohn et al., 2015), sample a latent variable $\boldsymbol{z}$ from two normal distributions, respectively. Supervised by a discriminator with the function label, the latent variable encodes meaningful information to realize a sentence function. The latent variable, along with the decoder's state, is also used to control the type of a word in generation via the type controller. In the decoder, the final generation distribution is mixed by the type distribution which is obtained from the type controller. By this means, the latent variable encodes information not only from sentence function but also from word types, and in return, the decoder and the type controller can deal with the compatibility of realizing sentence function and information content in generation.

3.2 Encoder-Decoder Framework

The encoder-decoder framework has been widely used in language generation (Sutskever et al., 2014; Vinyals and Le, 2015). The encoder transforms the post sequence $X = x_1 x_2 \cdots x_n$ into hidden representations $\boldsymbol{H} = \boldsymbol{h}_1 \boldsymbol{h}_2 \cdots \boldsymbol{h}_n$, as follows:

$$\boldsymbol{h}_t = \mathbf{GRU}(\boldsymbol{e}(x_t), \boldsymbol{h}_{t-1}) \quad (2)$$

where $\mathbf{GRU}$ is gated recurrent unit (Cho et al., 2014), and $\boldsymbol{e}(x_t)$ denotes the embedding of the word x_t.

The decoder first updates the hidden states $\boldsymbol{S} = \boldsymbol{s}_1 \boldsymbol{s}_2 \cdots \boldsymbol{s}_m$, and then generates the target sequence $Y = y_1 y_2 \cdots y_m$ as follows:

$$\boldsymbol{s}_t = \mathbf{GRU}(\boldsymbol{s}_{t-1}, \boldsymbol{e}(y_{t-1}), \boldsymbol{cv}_{t-1}) \quad (3)$$

$$y_t \sim P(y_t | y_{<t}, \boldsymbol{s}_t) = softmax(\boldsymbol{W}\boldsymbol{s}_t) \quad (4)$$

where this $\mathbf{GRU}$ does not share parameters with the encoder's network. The context vector $\boldsymbol{cv}_{t-1}$ is a dynamic weighted sum of the encoder's hidden states, i.e., $\boldsymbol{cv}_{t-1} = \sum_{i=1}^{n} \alpha_i^{t-1} \boldsymbol{h}_i$, and α_i^{t-1} scores the relevance between the decoder's state $\boldsymbol{s}_{t-1}$ and the encoder's state $\boldsymbol{h}_i$ (Bahdanau et al., 2015).

3.3 Recognition/Prior Network

On top of the encoder-decoder structure, our model introduces the recognition network and the prior network of CVAE framework, and utilizes the two networks to draw latent variable samples during training and test respectively. The latent variable can project different sentence functions into different regions in a latent space, and also capture various word patterns within a sentence function.

In the training process, our model needs to sample the latent variable from the posterior distribution $P(\boldsymbol{z}|Y, \boldsymbol{c})$, which is intractable. Thus, the recognition network $q_\phi(\boldsymbol{z}|Y, \boldsymbol{c})$ is introduced to approximate the true posterior distribution so that we can sample $\boldsymbol{z}$ from this deterministic parameterized model. We assume that z follows a multivariate Gaussian distribution whose covariance matrix is diagonal, i.e., $q_\phi(\boldsymbol{z}|Y, \boldsymbol{c}) \sim \mathcal{N}(\mu, \sigma^2 \mathbf{I})$. Under this assumption, the recognition network can be parameterized by a deep neural network such as a multi-layer perceptron (MLP):

$$[\mu, \sigma^2] = \mathbf{MLP}_{posterior}(Y, \boldsymbol{c}) \quad (5)$$

During test, we use the prior network $p_\theta(\boldsymbol{z}|\boldsymbol{c}) \sim \mathcal{N}(\mu', \sigma'^2 \mathbf{I})$ instead to draw latent variable samples, which can be implemented in a similar way:

$$[\mu', \sigma'^2] = \mathbf{MLP}_{prior}(\boldsymbol{c}) \quad (6)$$

To bridge the gap between the recognition and the prior networks, we add the KL divergence term that should be minimized to the loss function:

$$\mathcal{L}_1 = KL(q_\phi(\boldsymbol{z}|Y, \boldsymbol{c}) || p_\theta(\boldsymbol{z}|\boldsymbol{c})) \quad (7)$$

3.4 Discriminator

The discriminator supervises $\boldsymbol{z}$ to encode function-related information in a response with supervision signals. It takes $\boldsymbol{z}$ as input instead of the generated response Y to avoid the vanishing gradient of z, and predicts the function category conditioned on $\boldsymbol{z}$:

$$P(l|\boldsymbol{z}) = softmax(\boldsymbol{W}_D \cdot \mathbf{MLP}_{dis}(\boldsymbol{z})) \quad (8)$$

This formulation can enforce $\boldsymbol{z}$ to capture the features of sentence function and enhance the influence of $\boldsymbol{z}$ in word generation. The loss function of the discriminator is given by:

$$\mathcal{L}_2 = -E_{q_\phi(\boldsymbol{z}|Y,\boldsymbol{c})}[\log P(l|\boldsymbol{z})] \quad (9)$$

3.5 Type Controller

The type controller is designed to deal with the compatibility issue of controlling sentence function and generating informative content. As aforementioned, we classify the words in a response into three types: *function-related, topic, and ordinary* words. The type controller estimates a distribution over the word types at each decoding position, and the type distribution will be used in the mixture model of the decoder for final word generation. During the decoding process, the decoder's state $\boldsymbol{s}_t$ and the latent variable $\boldsymbol{z}$ are taken as input to estimate the type distribution as follows:

$$P(wt|\boldsymbol{s}_t, \boldsymbol{z}) = softmax(\boldsymbol{W}_0 \cdot \mathbf{MLP}_{type}(\boldsymbol{s}_t, \boldsymbol{z})) \quad (10)$$

Noticeably, the latent variable z introduced to the RNN encoder-decoder framework often fails to learn a meaningful representation and has little influence on language generation, because the RNN decoder may ignore z during generation, known as the issue of vanishing latent variable (Bowman et al., 2016). By contrast, our model allows $\boldsymbol{z}$ to directly control the word type at each decoding position, which has more influence on language generation.

3.6 Decoder

Compared with the traditional decoder described in Section 3.2, our decoder updates the hidden state $\boldsymbol{s}_t$ with both the input information $\boldsymbol{c}$ and the latent variable $\boldsymbol{z}$, and generates the response in a *mixture* form which is combined with the type distribution obtained from the type controller:

$$\boldsymbol{s}_t = \mathbf{GRU}(\boldsymbol{s}_{t-1}, \boldsymbol{e}(y_{t-1}), \boldsymbol{cv}_{t-1}, \boldsymbol{c}, \boldsymbol{z}) \quad (11)$$

$$P(y_t|y_{<t}, \boldsymbol{c}, \boldsymbol{z}) = P(y_t|y_{t-1}, \boldsymbol{s}_t, \boldsymbol{c}, \boldsymbol{z})$$
$$= \sum_{i=1}^{3} P(wt = i|\boldsymbol{s}_t, \boldsymbol{z}) P(y_t|y_{t-1}, \boldsymbol{s}_t, \boldsymbol{c}, \boldsymbol{z}, wt = i) \quad (12)$$

where $wt = 1, 2, 3$ stand for function-related words, topic words, and ordinary words, respectively. The probability for choosing different word types at time t, $P(wt = i|\boldsymbol{s}_t, \boldsymbol{z})$, is obtained from the type controller, as shown in Equation (10). The probabilities of choosing words in different types are introduced as follows:

Function-related Word: Function-related words represent the typical words for each sentence function, e.g., *what* for interrogative responses, and *please* for imperative responses. To select the function-related words at each position, we simultaneously consider the decoder's state $\boldsymbol{s}_t$, the latent variable $\boldsymbol{z}$ and the function category l.

$$P(y_t|y_{t-1}, \boldsymbol{s}_t, \boldsymbol{c}, \boldsymbol{z}, wt = 1) = softmax(\boldsymbol{W}_1 \cdot [\boldsymbol{s}_t, \boldsymbol{z}, \boldsymbol{e}(l)]) \quad (13)$$

where $\boldsymbol{e}(l)$ is the embedding vector of the function label. Under the control of $\boldsymbol{z}$, our model can learn to decode function-related words at proper positions automatically.

Topic Word: Topic words are crucial for generating an informative response. The probability for selecting a topic word at each decoding position depends on the current hidden state s_t:

$$P(y_t|y_{t-1}, \boldsymbol{s}_t, \boldsymbol{c}, \boldsymbol{z}, wt = 2) = softmax(\boldsymbol{W}_2 \boldsymbol{s}_t) \quad (14)$$

This probability is over the topic words we predict conditioned on a post. Section 3.8 will describe the details.

Ordinary Word: Ordinary words play a functional role in making a natural and grammatical sentence. The probability of generating ordinary words is estimated as below:

$$P(y_t|y_{t-1}, \boldsymbol{s}_t, \boldsymbol{c}, \boldsymbol{z}, wt = 3) = softmax(\boldsymbol{W}_3 \boldsymbol{s}_t) \quad (15)$$

The generation loss of the decoder is given as below:

$$\mathcal{L}_3 = -E_{q_\phi(\boldsymbol{z}|Y,\boldsymbol{c})}[\log P(Y|\boldsymbol{z}, \boldsymbol{c})]$$
$$= -E_{q_\phi(\boldsymbol{z}|Y,\boldsymbol{c})}[\sum_t \log P(y_t|y_{<t}, \boldsymbol{z}, \boldsymbol{c})] \quad (16)$$

3.7 Loss Function

The overall loss $\mathcal{L}$ is a linear combination of the KL term $\mathcal{L}_1$, the classification loss of the discriminator $\mathcal{L}_2$, and the generation loss of the decoder $\mathcal{L}_3$:

$$\mathcal{L} = \alpha \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3 \quad (17)$$

We let α gradually increase from 0 to 1. This technique of *KL cost annealing* can address the optimization challenges of vanishing latent variables in the RNN encoder-decoder (Bowman et al., 2016).

3.8 Topic Word Prediction

Topic words play a key role in generating an informative response. We resort to pointwise mutual information (PMI) (Church and Hanks, 1990) for predicting a list of topic words that are relevant to a post. Let x and y indicate a word in a post X and its response Y respectively, and PMI is computed as follows:

$$PMI(x, y) = \log \frac{P(x, y)}{P(x)P(y)} \quad (18)$$

Then, the relevance score of a topic word to a given post $x_1 x_2 \cdots x_n$ can be approximated as follows, similar to (Mou et al., 2016):

$$REL(x_1, ..., x_n, y) \approx \sum_{i=1}^{n} PMI(x_i, y) \quad (19)$$

During training, the words in a response with high REL scores to the post are treated as topic words. During test, we use REL to select the top ranked words as topic words for a post.

4 Experiment

4.1 Data Preparation

We collected a Chinese dialogue dataset from Weibo [2]. We crawled about 10 million post-responses pairs. Since our model needs the sentence function label for each pair, we built a classifier to predict the sentence function automatically to construct large-scale labeled data. Thus,

[2] http://www.weibo.com

we sampled about 2,000 pairs from the original dataset and annotated the data manually with four categories, i.e., *interrogative, imperative, declarative* and *other*. This small dataset was partitioned into the training, validation, and test sets with the ratio of 6:1:1. Three classifiers, including LSTM (Hochreiter and Schmidhuber, 1997), Bi-LSTM (Graves et al., 2005) and a self-attentive model (Lin et al., 2017), were attempted on this dataset. The results in Table 1 show that the self-attentive classifier outperforms other models and achieves the best accuracy of 0.78 on the test set.

Model	Accuracy
LSTM	0.60
Bi-LSTM	0.75
Self-Attentive	0.78

Table 1: Accuracy of sentence function classification on the 2,000 post-response pairs.

We then applied the self-attentive classifier to annotate the large dataset and obtained a dialogue dataset with noisy sentence function labels[3]. To balance the distribution of sentence functions, we randomly sampled about 0.6 million pairs for each sentence function to construct the final dataset. The statistics of this dataset are shown in Table 2. The dataset[4] is available at http://coai.cs.tsinghua.edu.cn/hml/dataset.

Training	#Post	1,963,382	
	#Response	Interrogative	618,340
		Declarative	672,346
		Imperative	672,696
Validation	#Post	24,034	
	#Response	Interrogative	7,045
		Declarative	9,685
		Imperative	7,304
Test	#Post	6,000	

Table 2: Corpus statistics.

4.2 Experiment Settings

Our model was implemented with TensorFlow[5]. We applied bidirectional GRU with 256 cells to the encoder and GRU with 512 cells to the decoder. The dimensions of word embedding and function category embedding were both set to 100. We also set the dimension of latent variables to 128. The vocabulary size was set to 40,000. Stochastic gradient descent (Qian, 1999) was used to optimize our model, with a learning rate of 0.1, a decay rate of 0.9995, and a momentum of 0.9. The batch size was set to 128. Our codes are available at https://github.com/kepei1106/SentenceFunction.

We chose several state-of-the-art baselines, which were implemented with the settings provided in the original papers:

Conditional Seq2Seq (c-seq2seq): A Seq2Seq variant which takes the category (i.e., function type) embedding as additional input at each decoding position (Ficler and Goldberg, 2017).

Mechanism-aware (MA): This model assumes that there are multiple latent responding mechanisms (Zhou et al., 2017). The number of responding mechanisms is set to 3, equal to the number of function types.

Knowledge-guided CVAE (KgCVAE): A modified CVAE which aims to control the dialog act of a generated response (Zhao et al., 2017).

4.3 Automatic Evaluation

Metrics: We adopted *Perplexity (PPL)* (Vinyals and Le, 2015), *Distinct-1 (Dist-1)*, *Distinct-2 (Dist-2)* (Li et al., 2016a), and *Accuracy (ACC)* to evaluate the models at the content and function level. Perplexity can measure the grammaticality of generated responses. Distinct-1/distinct-2 is the proportion of distinct unigrams/bigrams in all the generated tokens, respectively. Accuracy measures how accurately the sentence function can be controlled. Specifically, we compared the prespecified function (as input to the model) with the function of a generated response, which is predicted by the self-attentive classifier (see Section 4.1).

Model	PPL	Dist-1	Dist-2	ACC
c-seq2seq	57.14	949/.007	5177/.041	0.973
MA	**46.08**	745/.005	2952/.027	0.481
KgCVAE	56.81	1531/**.009**	10683/.070	0.985
Our Model	55.85	**1833**/.008	**15586/.075**	**0.992**

Table 3: Automatic evaluation with perplexity (PPL), distinct-1 (Dist-1), distinct-2 (Dist-2), and accuracy (ACC). The integers in the Dist-* cells denote the total number of distinct n-grams.

Results: Our model has lower perplexity than c-seq2seq and KgCVAE, indicating that the model is comparable with other models in generating grammatical responses. Note that MA has the lowest perplexity because it tends to generate generic responses.

[3]Though the labels are noisy, the data are sufficient to train a generation model in practice.

[4]Note that we strictly obeyed the policies of Weibo and anonymized potential private information in dialogues. This dataset is strictly limited for academic use.

[5]https://github.com/tensorflow/tensorflow

Model	Interrogative			Declarative			Imperative		
	Gram.	Appr.	Info.	Gram.	Appr.	Info.	Gram.	Appr.	Info.
Ours vs. c-seq2seq	0.534	0.536	0.896*	0.630*	0.573*	0.764*	0.685*	0.504	0.893*
Ours vs. MA	0.802*	0.602*	0.675*	0.751*	0.592*	0.617*	0.929*	0.568*	0.577*
Ours vs. KgCVAE	0.510	0.626*	0.770*	0.546*	0.515*	0.744*	0.780*	0.521*	0.837*

Table 4: Manual evaluation results for different functions. The scores indicate the percentages that our model wins the baselines after removing tie pairs. The scores of our model marked with * are significantly better than the competitors (Sign Test, *p-value* < 0.05).

As for distinct-1 and distinct-2, our model generates remarkably more distinct unigrams and bigrams than the baselines, indicating that our model can generate more diverse and informative responses compared to the baselines.

In terms of sentence function accuracy, our model outperforms all the baselines and achieves the best accuracy of 0.992, which indicates that our model can control the sentence function more precisely. MA has a very low score because there is no direct way to control sentence function, instead, it learns automatically from the data.

4.4 Manual Evaluation

To evaluate the generation quality and how well the models can control sentence function, we conducted pair-wise comparison. 200 posts were randomly sampled from the test set and each model was required to generate responses with three function types to each post. For each pair of responses (one by our model and the other by a baseline, along with the post), annotators were hired to give a preference (win, lose, or tie). The total annotation amounts to $200\times3\times3\times3$=5,400 since we have three baselines, three function types, and three metrics. We resorted to a crowdsourcing service for annotation, and each pair-wise comparison was judged by 5 curators.

Metrics: We designed three metrics to evaluate the models from the perspectives of sentence function and content: *grammaticality* (whether a response is grammatical and coherent with the sentence function we prespecified), *appropriateness* (whether a response is a logical and appropriate reply to its post), and *informativeness* (whether a response provides meaningful information via the topic words relevant to the post). Note that the three metrics were separately evaluated.

Results: The scores in Table 4 represent the percentages that our model wins a baseline after removing tie pairs. A value larger than 0.5 indicates that our model outperforms its competitor. Our model outperforms the baselines significantly in most cases (Sign Test, with *p-value* < 0.05). Among the three function types, our model performs significantly better than the baselines when generating declarative and imperative responses. As for interrogative responses, our model is better but the difference is not significant in some settings. This is because interrogative patterns are more apparent and easier to learn, thereby all the models can capture some of the patterns to generate grammatical and appropriate responses, resulting in more ties. By contrast, declarative and imperative responses have less apparent patterns whereas our model is better at capturing the global patterns through modeling the word types explicitly.

We can also see that our model obtains particularly high scores in informativeness. This demonstrates that our model is better to generate more informative responses, and is able to control sentence functions at the same time.

The annotation statistics are shown in Table 5. The percentage of annotations that at least 4 judges assign the same label (at least 4/5 agreement) is larger than 50%, and the percentage for at least 3/5 agreement is about 90%, indicating that annotators reached a moderate agreement.

	At least 3/5	At least 4/5
Grammaticality	91.7%	60.1%
Appropriateness	88.6%	52.5%
Informativeness	95.9%	71.2%

Table 5: Annotation statistics. *At least n/5* means there are no less than n judges assigning the same label to a record during annotation.

4.5 Words and Patterns in Function Control

To further analyze how our model realizes the global control of sentence function, we presented frequent words and frequent word patterns within each function. Specifically, we counted the frequency of a function-related word in the generated responses. The type of a word is predicted by the type controller. Further, we replaced the

Function	Frequent Words		Frequent Patterns		Response Examples	
	Chinese	English	Chinese	English	Chinese	English
Interrogative	? 是 吗 说 什么	? be particle mean what	x是说y吗？	Does x mean y?	你是说我帅吗？	Do you mean I'm handsome?
			x是在y吗？	Is x y?	你是在夸我吗？	Are you praising me?
			x在哪y啊？	Where does x y?	你在哪上班啊？	Where do you work?
			x想y什么z？	What z does x want to y?	你想要什么类型的？	What type do you want to choose?
Imperative	! 要 可以 来 请	! will can come please	那就y吧	Do y, then.	那就好好养着吧	Take care of yourself, then.
			x要把y给z	Let x give y to z.	我要把你的房子给你	Let me give your house to you.
Declarative	是 也 觉得 可是 没	be also/too think but no	x也是y，可是z	x also y, but z	我也是这么想的，可是我要找一个人，哈哈	I also think so, but I will find a person. Ha-ha.
			x也是，a都b	x, too, and a has b.	我也是，我的粉丝都被我震精了	Me, too, and my fans have been shocked by me.

Figure 3: Frequent function-related words and frequent patterns containing at least 3 function-related words. The letters denote the variables which replace ordinary and topic words in the generated responses. The underlined words in responses are those occurring in patterns.

ordinary and topic words of a generated response with variables and treated each response as a sequence of function-related words and variables. We then used the Apriori algorithm (Agrawal and Srikant, 1994) to mine frequent patterns in these sequences. We retained frequent patterns that consist of at most 5 words and appear in at least 2% of the generated responses.

Figure 3 presents the most frequent words (the second and third columns) and patterns (the fourth and fifth columns) for each function type. Note that the word patterns can be viewed as an abstract representation of sentence function. We observed that:

First, function-related words are distributed at multiple positions of a sentence, indicating that realizing a sentence function needs a global control by not only predicting the word types but also planning the words of different types properly.

Second, the frequent words clearly reveal the difference between function types. For instance, interrogatives like 什么(what), ？(?) and 吗(particle) are commonly seen in interrogative responses, words like 请(please), 来(come), and 要(will) occur frequently in imperative responses. Further, word patterns in different function types differ significantly (see the fourth/fifth columns), indicating that the model is able to learn function-specific word patterns.

Third, interrogative and imperative responses have explicit patterns, while declarative responses are more implicit and divergent in pattern. Interrogative responses fall into *Yes-No*, *Wh-*, *How-*, and other questions. Imperative responses generally start with the base form of verbs or imperative words including 请(please). Our model succeeds in capturing two typical forms in declarative responses: adversative and progressive structures.

4.6 Case Study

Post:	如果有一天我突然跟你绝交，你会怎么样？ What would you do if I suddenly broke up with you someday?
Target Function:	Interrogative
c-seq2seq:	你是谁？ Who are you?
MA:	我会彻查关于法律相关司法措施的结论吗？ Should I investigate the conclusion of judicial measure related to the law?
KgCVAE:	你是不是想我了 Did you miss me?
Our Model:	你**说**的**是**我的错**吗？** **Do** you **mean** that it's my fault**?**
Target Function:	Imperative
c-seq2seq:	我会好好的！ I'll be fine!
MA:	绝交基佬，绝交基佬 Break up with guys. Break up with guys.
KgCVAE:	我会告诉你的！ I will tell you!
Our Model:	我**会把**你的礼物**给**你的，**放心** **Rest assured** that I **would give** your gift **to** you.
Target Function:	Declarative
c-seq2seq:	我也是这么认为的，我觉得我是个好孩子 I think so, too. I feel that I'm a good man.
MA:	我也不知道怎么办，但是我怕怕 I don't know what to do, but I'm in fear.
KgCVAE:	我也觉得我是个好男人。 I think I'm a good man, too.
Our Model:	我**觉得**我**会**很傻，我**会**被我妈骂的 I **would think** that I was stupid and I **would** be blamed by my mother.

Figure 4: Generated responses of all the models for different sentence functions. In the responses of our model, function-related words are in **red** and topic words in blue. The word type is predicted by the type controller.

We presented an example in Figure 4 to show that our model can generate responses of different function types better compared to baselines. We can see that each function type can be realized by a natural composition of function-related words (in red) and topic words (in blue). Moreover, function-related words are different and are placed at different positions across function types, indicating that the model learns function-specific word patterns. These examples also show that the compatibility issue of controlling sentence function and generating informative content is well addressed by planning function-related and topic words properly.

Post	如果有一天我突然跟你绝交，你会怎么样？ What would you do if I suddenly broke up with you someday?
Interrogative Response #1	你**说**的**是**我的错**吗？** **Do** you **mean** that it's my fault**?**
Interrogative Response #2	你**会不会**说话**？** **Can** you speak normally**?**
Interrogative Response #3	你想我**怎样？** 我**要不要**绝交**？** **What do** you think I should do**? Shall** I break up with you**?**

Figure 5: Different patterns of interrogative responses generated by our model.

Furthermore, we verified the ability of our model to capture fine-grained patterns within a sentence function. We took interrogative responses as example and obtained responses by drawing latent variable samples repeatedly. Figure 5 shows interrogative responses with different patterns generated by our model given the same post. The model generates several Yes-No questions led by words such as 吗(do), 会(can) and 要(shall), and a Wh-question led by 怎样(what). This example shows that the latent variable can capture the fine-grained patterns and improve the diversity of responses within a function.

5 Conclusion

We present a model to generate responses with both controllable sentence function and informative content. To deal with the global control of sentence function, we utilize a latent variable to capture the various patterns for different sentence functions. To address the compatibility issue, we devise a type controller to handle function-related and topic words explicitly. The model is thus able to control sentence function and generate informative content simultaneously. Extensive experiments show that our model performs better than several state-of-the-art baselines.

As for future work, we will investigate how to apply the technique to multi-turn conversational systems, provided that the most proper sentence function can be predicted under a given conversation context.

Acknowledgments

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.

References

Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, pages 487–499.

Adrian Akmajian. 1984. Sentence types and the form-function fit. *Natural Language Linguistic Theory*, 2(1):1–23.

Nabiha Asghar, Pascal Poupart, Jesse Hoey, Xin Jiang, and Lili Mou. 2017. Affective neural response generation. *arXiv preprint arXiv:1709.03968*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of International Conference on Learning Representations*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Kyunghyun Cho, Bart Van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, pages 22–29.

Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 623–632.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104.

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 *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 634–642.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional lstm networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*, pages 799–804.

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

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Xiang Li, Lili Mou, Rui Yan, and Ming Zhang. 2016b. Stalematebreaker: A proactive content-introducing approach to automatic human-computer conversation. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 2845–2851.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of International Conference on Learning Representations*.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *Proceedings of 26th International Conference on Computational Linguistics*, pages 3349–3358.

Soichiro Murakami, Akihiko Watanabe, Akira Miyazawa, Keiichi Goshima, Toshihiko Yanase, Hiroya Takamura, and Yusuke Miyao. 2017. Learning to generate market comments from stock prices. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1374–1384.

Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151.

Laurie E. Rozakis. 2003. *The complete idiot's guide to grammar and style*. Alpha.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of AAAI conference on Artificial Intelligence*.

Iulian Vlad. Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of AAAI conference on Artificial Intelligence*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 1577–1586.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491.

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

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. In *International Conference on Machine Learning Deep Learning Workshop*.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Proceedings of AAAI conference on Artificial Intelligence*.

Zhou Yu, Ziyu Xu, Alan W Black, and Alex I. Rudnicky. 2016. Strategy and policy learning for non-task-oriented conversational systems. In *Proceedings of 17th Annual SIGdial Meeting on Discourse and Dialogue*.

George Yule. 2010. *The study of language*. Cambridge university press.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Ganbin Zhou, Ping Luo, Rongyu Cao, Fen Lin, Bo Chen, and Qing He. 2017. Mechanism-aware neural machine for dialogue response generation. In *Proceedings of AAAI conference on Artificial Intelligence*.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of AAAI conference on Artificial Intelligence*.

Xianda Zhou and William Yang Wang. 2017. Mojitalk: Generating emotional responses at scale. *arXiv preprint arXiv:1711.04090*.