

Category-Based Strategy-Driven Question Generator for Visual Dialogue

Yanan Shi¹, Yanxin Tan¹, Fangxiang Feng¹, Chunping Zheng¹ and Xiaojie Wang^{1✉}

¹Beijing University of Posts and Telecommunications, Beijing, China

apilis@bupt.edu.cn

tobytyx@gmail.com

{fxfeng, zhengchunping, xjwang}@bupt.edu.cn

Abstract

GuessWhat?! is a task-oriented visual dialogue task which has two players, a guesser and an oracle. Guesser aims to locate the object supposed by oracle by asking several Yes/No questions which are answered by oracle. How to ask proper questions is crucial to achieve the final goal of the whole task. Previous methods generally use an word-level generator, which is hard to grasp the dialogue-level questioning strategy. They often generate repeated or useless questions. This paper proposes a sentence-level category-based strategy-driven question generator(CSQG) to explicitly provide a category based questioning strategy for the generator. First we encode the image and the dialogue history to decide the category of the next question to be generated. Then the question is generated with the helps of category-based dialogue strategy as well as encoding of both the image and dialogue history. The evaluation on large-scale visual dialogue dataset GuessWhat?! shows that our method can help guesser achieve 51.71% success rate which is the state-of-the-art on the supervised training methods.

1 Introduction

Goal-oriented Visual Dialogue is one of the multi-modal task which has gained increasing attentions. It can usually be explained as a task which using a couple of visual related turns of communication to reach a specific goal, such as GuessWhich (Chattopadhyay et al., 2017), VisDial (Das et al., 2017), Multimodal dialogs (MMD) (Saha et al., 2018), GuessWhat?! (de Vries et al., 2017) and so-on.

GuessWhat?! (de Vries et al., 2017) is a two-player game which can be named guesser and oracle. Given an image, the oracle choose an object in the image without telling the guesser. Guesser reads the image and asks several Yes/No questions, at the meanwhile Oracle uses 'Yes', 'No' and 'N/A' to answer them. After a fixed turns of questioning, Guesser should guess the chosen object in image and one game finishes. As is mentioned above, there are two sub tasks: one is ask question(QGen), and the other is Guess. Figure 1 gives a sample in GuessWhat?! dataset.

To reach the goal of the entire game, the agent for task QGen should be able to make use of information from both the dialogue history and the image. de Vries et al. (de Vries et al., 2017) use an encoder-decoder model to generate questions, which is the first QGen model. After that, some researchers manage to improve the generation quality by attention mechanism. Zhuang et al. (Zhuang et al., 2018) proposed a parallel attention (PLAN) network. Deng et al. (Deng et al., 2018) proposed accumulated attention mechanism (A-ATT). Xu et al. (Xu et al., 2020) focused on boosting the agent to generate highly efficient questions and obtains reliable visual attentions and proposed Answer-Driven Visual State Estimator (ADVSE). On the other hand, Strub et al. (Strub et al., 2017) introduced reinforcement learning (RL). Pang et al. (Pang and Wang, 2019) proposed visual dialogue state tracking (VDST) to represent visual dialogue state and update with the change of the distribution on objects.

When humans play the game, they often have some strategies which aims to minimize the scope of the object and reaches the goal as quickly as possible. For example, In visual dialogue dataset GuessWhat?!, people might first ask the type of the object, once they determine the right type of the object, they then


	Questions	Answers
	is it a donut ?	Yes
	is it on the left ?	No
	is it on the right ?	Yes
	is it the whole donut ?	Yes
	does it have pink icing ?	No
	is it touching the donut with the sprinkles ?	Yes
	does it have chocolate icing ?	No
	is it the third donut?	Yes

Figure 1: This is a sample of visual dialogue in dataset GuessWhat?!. From the question sequence, we can see that human beings use the explicit category strategy to help locate the target.

ask the attributes of the object and so on. In fact, we can find there are more than 80% of records ask questions about an object at the first turn, however only 4.03% ask questions about color at the same place. The sample in dataset GuessWhat?! (de Vries et al., 2017) is shown in Figure 1. However, most of previous works didn't explicitly focus on the question generating strategy. de Vries et al. (de Vries et al., 2017) encodes the image and dialogue history into a fixed length representation vector and generates the response. Deng et al. (Deng et al., 2018) and Zhuang et al. (Zhuang et al., 2018) improve the model by adding strong attention mechanisms but still lack of improvement on generation side. In the absence of specialized strategy modules, such simple encoding way is often difficult to fit the question generation switching strategy (de Vries et al., 2017). Xu et al. (Xu et al., 2020) used an answer-based attention transfer method. They used the transferring of attention to shift the focus point of the problem, but still lacks of explicit influence on question generation.

We proposed a category-based strategy-driven question generator (CSQG) to solve the above issues. We design a module of category prediction before response generation. We first encode the image and historical information using faster-RCNN (Ren et al., 2016) and multi-layer GRU (Cho et al., 2014) respectively. Then we use text-image-based attention and combine the representation into context vectors. After that, we predict the probability distribution of the category of the question to be generated basing on the context vectors and a prior-probability of the category of next question, which is an explicit strategy on categories of questions learned from training data. Experimental results on GuessWhat?! (de Vries et al., 2017) show that our model effectively learns the information of category and achieve state-of-the-art performance on supervising learning domain.

In summary, our contributions are mainly as follows.

1. We propose a category-based strategy-driven question generator to explicitly introduce the strategy for question generation at dialogue level. An explicit prior-probability is used to control the category transfer together with encoding of the dialogue history and the image.
2. Experiments on large Visual Dialogue dataset GuessWhat?! show that our improved model achieves state-of-the-art performance on supervising learning setting.

In the rest of this paper, we firstly gives a general introduction to the relate work in section 2. Section 3 introduces the design of category information and how to label it on each question. Section 4 describes our CSQG model in detail. Section 5 shows the experiment results on large visual dialogue dataset and at last we gives our conclusion in section 6.

2 Related Work

Task-oriented visual dialogue is a new and critical research direction for dialogue systems. It needs not only the ability of traditional text-based dialogue (Chen et al., 2019; Shan et al., 2020; Tan et al., 2020), but also image representation and multi-modal fusion capabilities (Simonyan and Zisserman, 2014; Ren et al., 2016). As one of the widely used dataset, GuessWhat?! (de Vries et al., 2017) has three sub-tasks,

QGen, Oracle and Guesser. QGen is extensively studied due to its key role in the entire game. (de Vries et al., 2017; Xu et al., 2020; Pang and Wang, 2019).

There are mainly two lines of researches on QGen task. One line is the improvement on model structure. de Vries et al. (de Vries et al., 2017) proposed the first model for the task as a baseline. It uses the fc8-layer of VGG-16 model (Simonyan and Zisserman, 2014) to provide image embedding and a hierarchical RNN-based model to encode the dialogue history. The generator is performed by RNN-based model which concatenates image and context representation and generates responses. In order to have more clear image representation, ResNet (He et al., 2016) is introduced by the work from Shekhar et al. (Shekhar et al., 2018b) which extracts the image representation by the second to last layer of ResNet152. Due to the lack representation capability of static embedding, object detection models are involved. Xu et al. (Xu et al., 2020) and Pang et al. (Pang and Wang, 2019) introduced Faster-RCNN (Ren et al., 2016) and receive a better performance. Many of other researches focus on the improvement of linguistic information encoding. Other than the baseline method, some attention-based models are proposed to get dynamic visual and linguistic embedding. PLAN network (Zhuang et al., 2018) is proposed to provide dynamic visual information at each round by computing the attention on different regions. Accumulated Attention (Deng et al., 2018) consists of three kinds of attention which are query, image and objects attention. Answer-Driven Visual State Estimator (Xu et al., 2020) uses an answer-based attention transfer method to exploit different answers in different ways to update the visual attention at each step.

The other line is on model learning. Strub et al. (Strub et al., 2017) first used reinforce learning on this task. Temperature Policy Gradient (Zhao and Tresp, 2018) is proposed to make balance of exploration and exploitation while selecting words. Visual dialogue state tracking (VDST) (Pang and Wang, 2019) is proposed to update the change of the distribution on objects.

Our work is basically on the first line. Inspired by human cognitive psychology (Anderson and Crawford, 1980). and current controlled text generation tasks (Smith et al., 2020), we add control factors to influence the direction of text generation. To prepare questions controlled information, we first classify them like Krishna et al. (Krishna et al., 2019) at the earliest for VQA task. Different from them, we merge the unreasonable classifications and used a more accurate model-based method instead of keywords to label questions' categories.

In QGen task, our CSQG predicts the probability distribution of category and determines the certain category in this turn based on the explicit prior-probability strategy. As we can see, there is currently no work to add the question category as an element to the generation process. Our method is based on human cognition and has better interpretability and logic. The specific effect will also be confirmed in subsequent experiments.

3 Proposed model

In this section, we will introduce our question generation model CSQG in detail. Denote the dialogue history as H , $H_{t-1} = [q_0, a_0, q_1, a_1 \dots q_{t-1}, a_{t-1}]$ as history before the current turn, and I as the image representation. Our model can be described as $q_t = f(H_{t-1}, I)$ where q_t is the generated question for user in this turn. The overall structure of the model is shown in Figure 2. There are three module: (1) a multi-modality encoder with a hierarchical RNN-based neural network and an image-text attention structure, (2) a category predictor and (3) a generation decoder.

3.1 The design and implement of category info to questions

In the current most datasets (Saha et al., 2018; de Vries et al., 2017), there is no specific type annotation for each question. Therefore, before introducing the category-driven policy into visual dialogue generation task, we need to complete the classification and annotation of question categories. As one of our references, Krishna et al. (Krishna et al., 2019) defines 15 category tags (such as objects, attributes and so on) and marks each question with one or more of these tags in the vqa dataset (Antol et al., 2015). We find that 15 tags are too complex and unnecessary for our task. Thus we merge the tags and propose four categories: object, color, location and other.

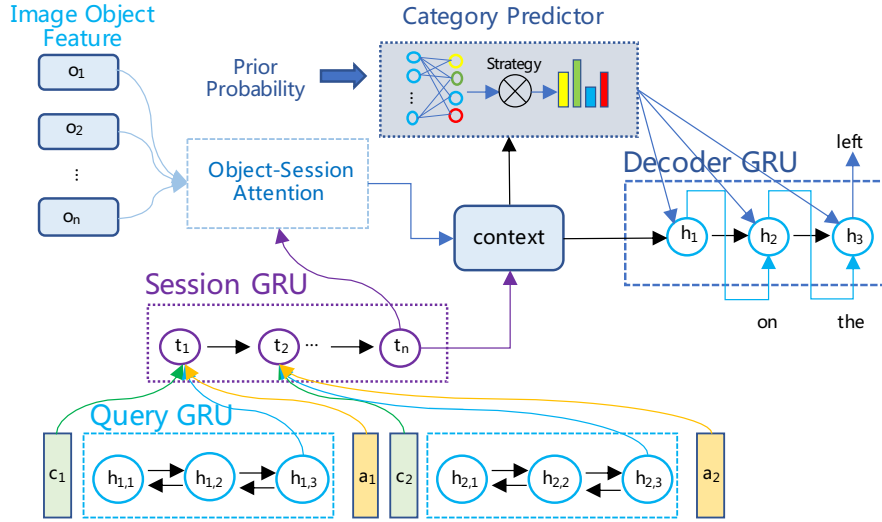


Figure 2: This is the structure of our CSQG model. The query and session GRU is hierarchical RNN-based dialogue history encoder. For the image object feature, we use object-session attention as shown in the top left of the figure. We then predict the next category and add it to the decoder as shown in the top and right of the figure.

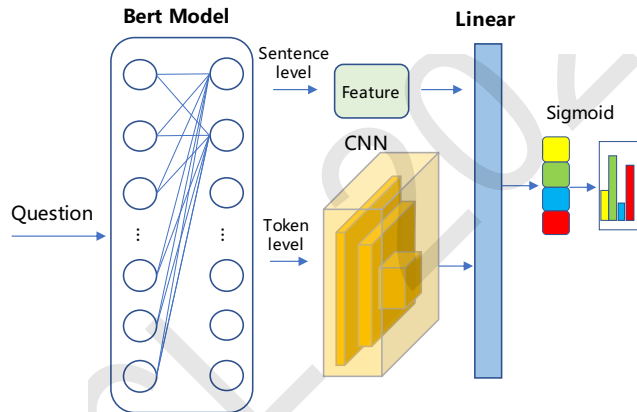


Figure 3: The Bert-CNN model for classifying category of questions.

The method that Krishna et al. (Krishna et al., 2019) tag questions is using key-words by counting the static words of each category. When those words appear in a question, it can be marked with the category tag. This method is simple but not accurate. Especially the key-words of location category are confused with many preposition words in normal questions.

As a result, we propose a supervised Bert-CNN classification model to deal the task which will be at first trained by a part of manual marking data. The model is shown in Figure 3.

Denote question Q with n tokens as $Q = [x_0, x_1, \dots, x_{n-1}]$. We first use pretrained Bert model (Devlin et al., 2018) to extract and get the representation of the sentence. When a list of question tokens is fed into the model, each token will be converted as word embedding $E_{word} \in R^h$ and position embedding $P_{word} \in R^d$ where d is the hidden dimension of the model. The embedding input sequence $[e_0, e_1, \dots, e_{n-1}]$ is then fed into Bert model which is basically a Bi-Transformer encoder containing several blocks, from which the main structure is Multi-head Self-attention (Vaswani et al., 2017). The final output of Bert model can be expressed as below:

$$BertInput(x_i) = E(x_i) + P(x_i) = e_i \in R^d, \forall 0 \leq i \leq n \quad (1)$$

$$BertOut = BertModel([e_0, e_1, \dots, e_{n-1}]) = [t_0, t_1, \dots, t_{n-1}] \quad (2)$$

We split the output into token-level and sentence-level representation. Sentence-level representation $O_{sentence}$ is the final state corresponding to the [CLS] token which we added in the beginning of input sentence. Token-level representation O_{tokens} is the final state corresponding to each token in the raw sentence. A convolution layer is added behind to deal with token-level representation for more classification information.

Concatenate the convolution output and the sentence-level representation from Bert model, we map features to specific categories through a full-connected layer and use the Sigmoid function to normalize the probability of each category.

In the face of millions of questions, it is not realistic to label all the question categories manually. Therefore, we estimate the minimum number of training data required by our model, and then labeling them manually. After the model is trained completely, we label the category information of the remaining questions by inference process of this model.

3.2 Multi-Modality Encoder

The multi-modality encoder is used to get the representation from current context which include the image feature and the dialogue history. The dialogue history contains several turns of question-answer pairs which can be expressed as $D = [q_0, a_0, q_1, a_1, \dots, q_{n-1}, a_{n-1}]$. Each question is an independent sentence while the answer is limited into three choices: 'Yes', 'No', 'N/A'. Each question and its corresponding answer will form a turn of game, and there is an obvious progressive relationship between each turn. So combine all the questions and answers into a long list is not a wise approach. Refer to Serban et al. (Serban et al., 2016), we build a hierarchical recurrent neural network based on GRU (Cho et al., 2014) as our dialogue history encoder. It can be divided as two layers.

The first layer is query layer, which is Bi-GRU. It process each question and output the last hidden. Suppose the i -th question sentence is $q_i = [x_{i,0}, x_{i,1}, \dots, x_{i,n-1}]$, where $x_{i,j} \in R^{d_{emb}}$ means the embedding of j -th token in question sequence q_i . The procedure of each timestamp is shown below:

$$h_n^{forward} = GRU(x_n, h_{n-1}, W) \quad (3)$$

$$h_n^{backward} = GRU(x_n, h_{n+1}, W) \quad (4)$$

$$h_n = h_n^{forward} + h_n^{backward} \quad (5)$$

$h_{n-1} \in R^{d_{query}}$ means the hidden state of last timestamp, $h_{n+1} \in R^{d_{query}}$ means the hidden state of next timestamp, and $W \in R^{d_{emb}, d_{query}}$ means the parameters in this layer. When the last token embedding is processed, we summed each backward and forward hidden state and the last timestamp's hidden state is regarded as the representation of this sequence.

After all the questions are encoded, our model needs to further encode the previous history messages at the session level. The second layer is session layer. Not like the former query layer, this is based on Uni-GRU with only forward direction. For turn i , the input information contains question q_i , answer and the category. Because the types of both answer and category are limited, we transfer them by using one-hot embedding and express answer and category embedding of as a_i and c_i . We concatenate them and input to the session layer.

On the other hand, we introduce the commonly used object detection model Faster-RCNN (Ren et al., 2016) and get the feature of objects in the image. Besides the representation vector, we concatenate the related position from each object to others. The final object features of image I is expressed as below. R_{pos} is the relative position matrix and num is the number of objects in the image.

$$O_I = Concat(RCNN(I), R_{pos}) \in R^{num \times d_{image}} \quad (6)$$

Considering that the attention of different objects is different in each turn, we introduces the object-session attention mechanism to guide the attention of different objects. For object features list O_I and history representation t_n , a dot-attention is introduced to calculate the similarity score.

$$v_{mid} = Conv1d(O_I) \in R^{num \times d_{mid}} \quad (7)$$

$$t_{mid} = unsqueeze(W \cdot t_n) \in R^{1 \times d_{mid}} \quad (8)$$

$$h_{att} = t_{mid} \cdot v_{mid}^T \in R^{1 \times num} \quad (9)$$

In equation 9, h_{att} is the score of each object. After softmax function, we sum all objects weighted and concatenate with history representation t_n to get the context vector C as shown in equation 11.

$$V_I = \sum_{i=0}^{num-1} Softmax(h_{att}) * O_i \quad (10)$$

$$C = concat(V_I, t_n) \in R^{d_{context}} \quad (11)$$

3.3 Category Predictor

Category predictor is used to predict the category of the next question to guide the generation. Because we use the explicit way to express the category, the prediction task can be regarded as a multi-labeled classification task. The original method is using a full-connected neural network to get the signal of each category from the context vector and turn it to probability by a sigmoid function.

$$s_i = W^s \cdot C + b^s \quad (12)$$

$$p_i = Sigmoid(s_i) \quad (13)$$

W^s and b^s are parameters for full-connected neural network, and p_i is the probability of category i . When p_i exceeds the threshold, category i is considered to appear in the next question. In the training process, we directly use binary cross-entropy(BCE) loss function to obtain gradient information as below, where y_i is the target of category label i . After backward propagation of gradient and iteration of parameters by optimizer, the predictor will be able to forecast the needed category in the following generation task.

$$Loss = \sum_{i=1}^C y_i * \log p_i + (1 - y_i) * \log(1 - p_i) \quad (14)$$

In addition to simply predicting categories through neural networks, we introduce an explicit prior-probability strategy based on the answers.

When generating questions, there is an interesting phenomenon that the historical answers usually have an important impact on the category of questions to be generated in this turn. Especially the answer from last turn will have a direct and significant impact.

Considering that when the last question receive a positive answer, they tend to change the category of questions to expect more information gain. When the answer has a disposition to negative, the next question can continue to search for other possibilities in this domain. Based on the above, we make a constraint between the last turn's answer & category and the forecast category. The flow chart of the strategy is shown in figure 4. In addition to using the classification model to obtain the initial probability value $p_{t,i}$ of category i at turn t , we make statistics on the prior probability in the game. As the prior probability of same category class varies greatly between different turns, we separate them where $q_{t,i}$ means the prior probability of category i in t -th turn. This prior probability is used as benchmark to judge the confidence of predictor result.

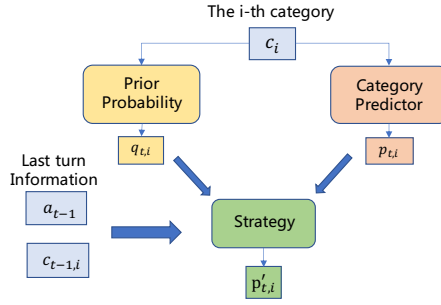


Figure 4: This is the strategy for generating next question’s category. In the figure, c_i means the i -th category. a_{t-1} and $c_{t-1,i}$ means the answer and i -th category of last turn. $p'_{t,i}$ is the final probability for the following generation task.

The principle of discriminator is described in equation 15:

$$p'_{t,i} = \begin{cases} \min(p_t) & (p_{t,i} < q_{t,i}) \& (a_{t-1} = Yes) \& (c_{t-1,i} = 1) \\ p_{t,i} & else \end{cases} \quad (15)$$

a_{t-1} and $c_{t-1,i}$ means the answer and i -th category of last turn. When last turn’s question has involved such category and answer is ”Yes”, the category’s income is greatly reduced. If predictor gives a high enough confidence which exceeds a given threshold (its prior probability in this turn), we trust it and continue to use $p_{t,i}$. Otherwise, we cut down its probability to the minimum.

For example, the category of last turn is B, and the answer is ”Yes”. The predicted probability of all categories is [A, B, C, D]. For category at position B, if B is greater than its prior probability, we adopt B. If B is lower than the prior probability, we cut down B to the minimum between A D to transfer the attention to other categories.

3.4 Generation Decoder

The generation decoder is used to generate questions based on context representation and predicted category. Its structure is based on GRU. Similar to common LSTM based sequence-to-sequence models (Dušek and Jurčiček, 2016; Sutskever et al., 2014), the first hidden state of decoder GRU is the context representation C computed in equation 11. During each timestamp, we combine the input embedding and the category probability vector to guide the decoder generating preset types of tokens. Suppose the input for GRU is $input_t^{dec}$ which is concatenated by category probability vector p_c and token embedding emb_j^{dec} . The hidden state of last generated token is h_{j-1}^{dec} and the computed progress is as below:

$$h_j^{dec} = GRU(h_{j-1}^{dec}, input_t^{dec}) \quad (16)$$

$$logits_j = W_{lm}^{dec} \cdot h_j^{dec} \quad (17)$$

The outputs of each timestamp are updated hidden state h_j^{dec} and current output logits $logits_j$. Cross entropy is used to compute loss and generate gradient. And in inference procedure, beam search (Freitag and Al-Onaizan, 2017) is the generating strategy we use to maximize the general probability of the whole question.

4 Experiments

To access the performance, we evaluate our models on the large visual dialogue dataset GuessWhat?!, which contains 155,281 dialogues and 821,955 question-answer pairs. These dialogues are based on 66537 images with 134,074 different objects. To make a fair contrast between ours and the previous, we divided GuessWhat?! dataset according to the same way from others. All dialogues are split into train, dev and test dataset, 70%, 15%, 15%.

In this section, we show the performance of CSQG on the question generation task. The implement details of module Oracle and Guesser will be introduced, and we will report the contrast experiment between ours and related comparison objects. In order to analyze the effect in more detail, an ablation experiment, the accuracy of category in generating questions and representative case analysis will be reported in the next.

4.1 Implementation Details on Guesser and Oracle

There are three parts in a GuessWhat?! game in all. Besides question generation task, we implement the baselines of Oracle and Guesser module that assist to complete the whole game.

The job for an oracle is to answer questions based on the given object. We choose GRU to build a question encoder and the representation is the hidden state of last timestamp. The hidden size of GRU is 2400, with only forward direction and 300 embedding dimension. The representation of the object contains crop vector, spatial and object category information. After concatenate both features, a feed-forward neural network is built with two linear layer and a ReLU (Glorot et al., 2011) activation function beside them. The middle hidden size is 512 and the output is the score of each kind of answers. After a Softmax function, we compute the loss by cross entropy function. After 15 epochs with batch size of 128, the accuracy on the test set is about 78.5%.

The job for an guesser is to guess which object has been selected in advance according to the information of dialogues, image and objects. The dialogue encoder is also built by GRU with only forward direction. Different from Oracle, the image feature is extracted from the fc8 layer of pretrained vgg16 model (Simonyan and Zisserman, 2014). A feed-forward neural network is used to transfer the object features into vectors, whose are computed cosine distances with dialogue hidden state. The feed-forward neural network contains two linear layer with 512 hidden size and ReLU activation function. At last we softmax the scores and guess the object. The accuracy of this baseline is about 64.6%.

4.2 Contrast Experiments

Follow the existing studies (de Vries et al., 2017; Shekhar et al., 2018a; Zhang et al., 2018), we choose the comprehensive game success rate as evaluation metrics. The reason we don't choose traditional text generation evaluation indicators such as Bleu (Papineni et al., 2002) and Rouge (Lin, 2005) is that the diversity of questions will affect the accuracy of these evaluation metrics.

We compared the recent representative models in this field, whose are baseline SL (de Vries et al., 2017), VDST-SL (Pang and Wang, 2019), ADVSE-QGen (Xu et al., 2020) and GDSE-SL (Shekhar et al., 2018b). Because the implementation of CSQG is only under supervised learning, we compare with other models only on supervised learning part.

Baseline SL (de Vries et al., 2017): This is an end-to-end QGen model with HRED dialogue encoder and decoder. It uses the output of fc8 layer of VGG16 as image feature.

VDST-SL (Pang and Wang, 2019): An end-to-end QGen model based on visual dialogue state tracking structure. It includes a visual-language-visual multi-step reasoning cycle. Refer to the description from Pang et al (Pang and Wang, 2019), the dimension of word embedding is set to 512. We don't append the reinforce training part.

ADVSE-QGen (Xu et al., 2020) : An end-to-end model with answer-driven focusing attention, which gets visual state estimation by conditional visual information fusion.

GDSE-SL (Shekhar et al., 2018b): It contains a grounded dialogue state encoder which addresses a foundational issue on how to integrate visual grounding with dialogue system components. It is only trained by supervised learning.

CSQG-CD and CSQG are ours. CSQG-CD only includes category driven, without adding specific strategies while CSQG contains both category driven structure and prior probability strategy. We use the vector of objects from faster-RCNN which contains 36 objects per image and each size is 2048. The hidden size of our hierarchical encoder is 800 for query layer and 1000 for session layer. The hidden size of our decoder is 600. The dropout is set as 0.2 for the whole model.

Table 1 shows comparisons among the above models. We conducted comparative experiments on new targets which pictures have been seen by models and on new images which totally new for models. 5

Table 1: Comparison results of QGen on the task success rate. G means Greedy Search and B means Beam Search.

		New Object		New Game	
		Max Turn 5	Max Turn 8	Max Turn 5	Max Turn 8
baseline SL	G	43.5	-	40.8	40.7
	B	47.1	-	44.6	-
VDST-SL	G	49.49	48.01	45.94	45.03
ADVSE-QGen	G	50.66	-	47.03	-
	B	47.47	-	44.70	-
GDSE-SL	G	-	-	47.8	49.7
CSQG-CD(Ours)	G	52.4	52.7	48.0	48.3
	B	51.5	52.7	46.6	48.2
CSQG(Ours)	G	53.2	54.4	49.9	51.7
	B	52.4	53.9	48.1	49.7

Table 2: Ablation results of QGen on the task success rate. The generation strategy is greedy search.

	New Object		New Game	
	Max Turn 5	Max Turn 8	Max Turn 5	Max Turn 8
Baseline	43.9	44.3	41.6	41.8
CSQG-CD	52.4	52.7	48.0	48.3
CSQG-Punish	52.1	53.7	47.0	48.4
CSQG	53.2	54.4	49.9	51.7

and 8 turns of dialogue are set respectively, with both beam search and greedy search. CSQG achieves the success rate of 54.4% on new objects and 51.7% on new games. It exceeds 9.7% on new objects and 11% on new games from baseline-SL and 2% from the current state of the art GDSE-SL. Ours achieve the highest success rate under supervising learning at present.

The comparative experiments show that our module for controlling question generation category and the strategy of predicting categories effectively improve the key success rate of games. Although we only use the baseline modules of oracle and guess, the overall success rate of the game is still significantly improved. This shows that the conditional introduction of high-level semantic control information still has room for improvement compared with pure end-to-end models whose are actually pure black boxes.

4.3 Ablation Experiments

In order to evaluate efficiency of the different improvements in our models, we conduct ablation experiments on settings of category-driven structure, category strategy and prior probability of categories.

Baseline is an end-to-end QGen model with RCNN image features and visual attention between dialogue sessions and images. The reason why we don't use the model proposed by de Vries et al. (de Vries et al., 2017) is to eliminate the difference from image features of VGG16 and RCNN.

CSQG-CD includes category driven without adding specific strategies. The next question's category is entirely predicted by neural networks from the context vector.

CSQG-Punish is with different category prediction strategy. Instead of prior probability in each turn, it only punishes the probability when same former categories receive "Yes". The more positive answers, the more punishment for this category.

The rest of settings remain keep up with CSQG where the hidden sizes of encoder are 800 and 1000. The image feature is 36×2048 from RCNN and the visual attention is set with the hidden size of 800 and dropout of 0.2. The ablation experiment result shows in Table 2. RCNN-based baseline achieves 44.3% on new objects and 41.8% on new games.

The improvement of category-driven raises the success rate from 44.3% and 41.8% to 52.7% and

48.3%(comparing Baseline with CSQG-CD). It approves the positive impact of this improvement. Comparing with pure NN-predicted category method, the punishment strategy raises the success rate from 52.7% and 48.3% to 53.7% and 48.4%. This proves that compared with human questioning strategy, simply relying on neural networks can not make the global revenue maximization. Although we can rely on more complex reinforcement learning optimization, our strategy is obviously more concise and good interpretability. The comparison between CSQG-Punish and CSQG shows that prior probability of categories can adjust the strategy to gain the maximum incomes.

4.4 Good Case Study

Figure 5 shows a couple of examples of dialogues generated by CSQG, whose are sampled from the evaluation set. As can be seen in figure, the sample at the top generates questions according to the categories very effectively. The subsequent categories are adjusted accordingly with the answers. When the answer is "Yes", almost all the next turns have changed the categories, which is also reasonable from the case point of view. These cases proves that our prior probability category strategy is reasonable and well implemented.



	Categories	Questions	Answers
	q_object other location color location location color q_object	is it a person ? is it the one of the ball ? is it one of the one of the person ? is it the one with the blue shirt ? is it the next to the left ? is it the next to the right ? is it the whole person in the black shirt ? is it the whole person ?	Yes No Yes No No No Yes Yes
	q_object color color location other location location other	is it food ? is it green ? is it brown ? is it on the right ? does it have a slice ? is it the one of the piece ? is it the piece ? is it the piece ?	Yes No Yes Yes Yes Yes Yes Yes

Figure 5: Generated dialogue examples of CSQG.

5 Conclusions

We introduced CSQG, a category-based strategy-driven question generator which is guided with category information. At First, we predict the initial probability of each class by classifier and joint learning training method. Then we introduce an update strategy based on the prior probability of each category in each turn. The updated category guides the decoder to generate the next question. Experiments shows that our proposed model achieves 51.7% on new games in large visual dialogue dataset GuessWhat?!, which is the state of the art under supervise learning. The ablation experiments further proved that both category guided and prior probability have positive effect on the success rate of the game.

Our CSQG model is training only on supervised learning. However, due to the coherence, interactivity and dynamics of visual dialogue, there is a lot of room for improvement in reinforcement learning. In the future, we will explore how to produce category prediction which is more consistent with the overall goal through reinforcement learning. And this will balance the goals of interpretability and game success rate.

Acknowledgments

We would like to thank anonymous reviewers for their suggestions and comments. The work was supported by the National Natural Science Foundation of China (NSFC62076032) and the Cooperation Project with Beijing SanKuai Technology Co., Ltd.

References

- John Robert Anderson and Jane Crawford. 1980. *Cognitive psychology and its implications*. wh freeman San Francisco.
- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December.
- Prithvijit Chattopadhyay, Deshraj Yadav, Viraj Prabhu, Arjun Chandrasekaran, Abhishek Das, Stefan Lee, Dhruv Batra, and Devi Parikh. 2017. Evaluating Visual Conversational Agents via Cooperative Human-AI Games. *arXiv:1708.05122 [cs]*, August. arXiv: 1708.05122.
- Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.
- 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. *arXiv preprint arXiv:1406.1078*.
- Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M. F. Moura, Devi Parikh, and Dhruv Batra. 2017. Visual Dialog. *arXiv:1611.08669 [cs]*, August. arXiv: 1611.08669.
- Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. 2017. GuessWhat?! Visual object discovery through multi-modal dialogue. *arXiv:1611.08481 [cs]*, February. arXiv: 1611.08481.
- Chaorui Deng, Qi Wu, Qingyao Wu, Fuyuan Hu, Fan Lyu, and Mingkui Tan. 2018. Visual grounding via accumulated attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Ondřej Dušek and Filip Jurčiček. 2016. A context-aware natural language generator for dialogue systems. *arXiv preprint arXiv:1608.07076*.
- Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. *arXiv preprint arXiv:1702.01806*.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 315–323.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 2019. Information maximizing visual question generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2008–2018.
- C Lin. 2005. Recall-oriented understudy for gisting evaluation (rouge). Retrieved August, 20:2005.
- Wei Pang and Xiaojie Wang. 2019. Visual Dialogue State Tracking for Question Generation. *arXiv:1911.07928 [cs]*, November. arXiv: 1911.07928.
- 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 of the Association for Computational Linguistics*, pages 311–318.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149.
- Amrita Saha, Mitesh Khapra, and Karthik Sankaranarayanan. 2018. Towards Building Large Scale Multimodal Domain-Aware Conversation Systems. *arXiv:1704.00200 [cs]*, January. arXiv: 1704.00200.
- Iulian Serban, Alessandro Sordani, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

- Yong Shan, Zekang Li, Jinchao Zhang, Fandong Meng, Yang Feng, Cheng Niu, and Jie Zhou. 2020. A contextual hierarchical attention network with adaptive objective for dialogue state tracking. *arXiv preprint arXiv:2006.01554*.
- Ravi Shekhar, Tim Baumgartner, Aashish Venkatesh, Elia Bruni, Raffaella Bernardi, and Raquel Fernández. 2018a. Ask no more: Deciding when to guess in referential visual dialogue. *arXiv preprint arXiv:1805.06960*.
- Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2018b. Beyond task success: A closer look at jointly learning to see, ask, and guesswhat. *arXiv preprint arXiv:1809.03408*.
- Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Eric Michael Smith, Diana Gonzalez-Rico, Emily Dinan, and Y-Lan Boureau. 2020. Controlling style in generated dialogue. *arXiv preprint arXiv:2009.10855*.
- Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. 2017. End-to-end optimization of goal-driven and visually grounded dialogue systems. *arXiv preprint arXiv:1703.05423*.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.
- Yanxin Tan, Zhonghong Ou, Kemeng Liu, Yanan Shi, and Meina Song. 2020. Turn-level recurrence self-attention for joint dialogue action prediction and response generation. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, pages 309–316. Springer.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.
- Zipeng Xu, Fangxiang Feng, Xiaojie Wang, Yushu Yang, Huixing Jiang, and Zhongyuan Wang. 2020. Answer-Driven Visual State Estimator for Goal-Oriented Visual Dialogue. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 4271–4279, Seattle WA USA, October. ACM.
- Junjie Zhang, Qi Wu, Chunhua Shen, Jian Zhang, Jianfeng Lu, and Anton Van Den Hengel. 2018. Goal-oriented visual question generation via intermediate rewards. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 186–201.
- Rui Zhao and Volker Tresp. 2018. Learning goal-oriented visual dialog via tempered policy gradient. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 868–875. IEEE.
- Bohan Zhuang, Qi Wu, Chunhua Shen, Ian Reid, and Anton van den Hengel. 2018. Parallel attention: A unified framework for visual object discovery through dialogs and queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.