Conversational Graph Grounded Policy Learning for Open-Domain Conversation Generation

Jun Xu[1]*, **Haifeng Wang**[2], **Zheng-Yu Niu**[2], **Hua Wu**[2], **Wanxiang Che**[1]†, **Ting Liu**[1]
[1]Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, Harbin, China
[2]Baidu Inc., Beijing, China
{jxu, car, tliu}@ir.hit.edu.cn, {wanghaifeng, niuzhengyu, wu_hua}@baidu.com

Abstract

To address the challenge of policy learning in open-domain multi-turn conversation, we propose to represent prior information about dialog transitions as a graph and learn a graph grounded dialog policy, aimed at fostering a more coherent and controllable dialog. To this end, we first construct a conversational graph (CG) from dialog corpora, in which there are vertices to represent "what to say" and "how to say", and edges to represent natural transition between a message (the last utterance in a dialog context) and its response. We then present a novel CG grounded policy learning framework that conducts dialog flow planning by graph traversal, which learns to identify a what-vertex and a how-vertex from the CG at each turn to guide response generation. In this way, we effectively leverage the CG to facilitate policy learning as follows: (1) it enables more effective long-term reward design, (2) it provides high-quality candidate actions, and (3) it gives us more control over the policy. Results on two benchmark corpora demonstrate the effectiveness of this framework.

1 Introduction

How to effectively learn dialog strategies is an enduring challenge for open-domain multi-turn conversation generation. To address this challenge, previous works investigate word-level policy models that simultaneously learn dialog policy and language generation from dialog corpora (Li et al., 2016b; Zhang et al., 2018b). But these word-level policy models often lead to a degeneration issue where the utterances become ungrammatical or repetitive (Lewis et al., 2017). To alleviate this issue, utterance-level policy models have been proposed to decouple policy learning from response generation, and they focus on how to incorporate

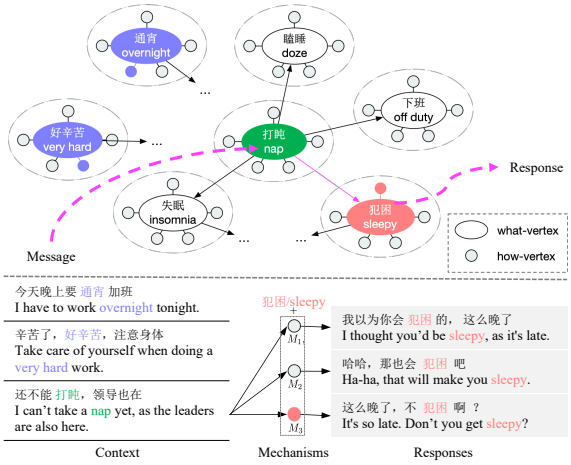

Figure 1: Our system (1) understands the user message by linking it to CG. We call the linked vertices as hit what-vertices (green color) ; (2) selects a what-vertex ("sleepy") and a how-vertex (responding mechanism M_3, a MLP network) from one-hop neighbors of hit vertices; (3) generates a coherent response with two sub-steps: firstly, obtains a response representation $\bar{r}$ using both M_3 and a message representation (from a message-encoder); Next, produces a response "It's so ..." with "sleepy" and $\bar{r}$ as input. Notice **all** the how-vertices are from the same set rather than completely independent of each other.

high-level utterance representations, e.g., latent variables or keywords, to facilitate policy learning (He et al., 2018; Yao et al., 2018; Zhao et al., 2019).

However, these utterance-level methods tend to produce less coherent multi-turn dialogs since it is quite challenging to learn semantic transitions in a dialog flow merely from dialog data without the help of prior information. In this paper, we propose to represent *prior information about dialog transition* (between a message and its response) as a graph, and optimize dialog policy based on the graph, to foster a more coherent dialog.

To this end, we propose a novel conversational graph (CG) grounded policy learning frame-

*This work was done at Baidu.
†Corresponding author: Wanxiang Che.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1835–1845
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

work for open-domain multi-turn conversation generation (**CG-Policy**). It consists of two key components, (1) a CG that captures both local-appropriateness and global-coherence information, (2) a reinforcement learning (RL) based policy model that learns to leverage the CG to foster a more coherent dialog. In Figure 1, given a user message, our system selects a what-vertex ("sleepy") and a how-vertex(responding mechanism M_3) to produce a coherent response.[1]

We first construct the CG based on dialog data. We use vertices to represent utterance content, and edges to represent dialog transitions between utterances. Specifically, there are two types of vertices: (1) a what-vertex that contains a keyword, and (2) a how-vertex that contains a responding mechanism (from a multi-mapping based generator in Section 3.1) to capture rich variability of expressions. We also use this multi-mapping based method to build edges between two what-vertices to capture the local-appropriateness between the two keywords as a message and a response respectively. It can be seen that the what-vertices from the same highly connected region are more likely to constitute coherent dialog.

We then present a novel graph grounded policy model to plan a long-term success oriented vertex sequence to guide response generation. Specifically, as illustrated by the three pink lines in Figure 1, given a user message, CG-Policy first links its keywords to CG to obtain hit what-vertices. Next, the policy model learns to select a what-vertex from one-hop what-vertex neighbors of all hit what-vertices, and then select a how-vertex from how-vertex neighbors of the chosen what-vertex. Finally, the two selected vertices are utilized to guide response generation. Thus we leverage the prior dialog-transition information (as graph edges) to narrow down candidate response content for more effective policy decision, instead of using the whole set of keywords as candidate actions. Moreover, to facilitate the modeling of long-term influence of policy decisions in an ongoing dialog, we first present novel CG based rewards to better measure the long-term influence of selected actions. We then employ a graph attention mechanism and graph embedding to encode global structure information of CG into dialog state representations, enabling global information aware decisions.

[1]Each mechanism is a MLP network to model how to express response content (Chen et al., 2019).

This paper makes the following contributions:

- This work is the first attempt that represents dialog transitions as a graph, and conducts graph grounded policy learning with RL. Supported by CG and this policy learning framework, CG-Policy can respond better in terms of local appropriateness and global coherence.
- Our study shows that: (1) one-hop what-vertex neighbors of hit what-vertices provide locally-appropriate and diverse response content; (2) the CG based rewards can supervise the policy model to promote a globally-coherent dialog; (3) the use of how-vertices in CG can improve response diversity; (4) the CG can help our system succeed in the task of target-guided conversation, indicating that it gives us more control over the dialog policy.

2 Related Work

Policy learning for chitchat generation To address the degeneration issue of word-level policy models (Li et al., 2016b; Zhang et al., 2018b), previous works decouple policy learning from response generation, and then use utterance-level latent variables (Zhao et al., 2019) or keywords (Yao et al., 2018) as RL actions to guide response generation. In this work, we investigate how to use prior dialog-transition information to facilitate dialog policy learning.

Knowledge aware conversation generation There are growing interests in leveraging knowledge bases for generation of more informative responses (Dinan et al., 2019; Ghazvininejad et al., 2018; Moghe et al., 2018; Zhou et al., 2018; Liu et al., 2019; Bao et al., 2019; Xu et al., 2020). In this work, we employ a dialog-modeling oriented graph built from dialog corpora, instead of a external knowledge base, in order to facilitate multi-turn policy learning, instead of dialog informativeness improvement.

Specifically, we are motivated by (Xu et al., 2020). The method in (Xu et al., 2020) has the issue of cross-domain transfer since it relies on labor-intensive knowledge graph grounded multi-turn dialog datasets for model training. Compared with them, our conversational graph is automatically built from dialog datasets, which introduces very low cost for training data construction. Furthermore, we decouple conversation modeling into two parts: "what to say" modeling and "how to

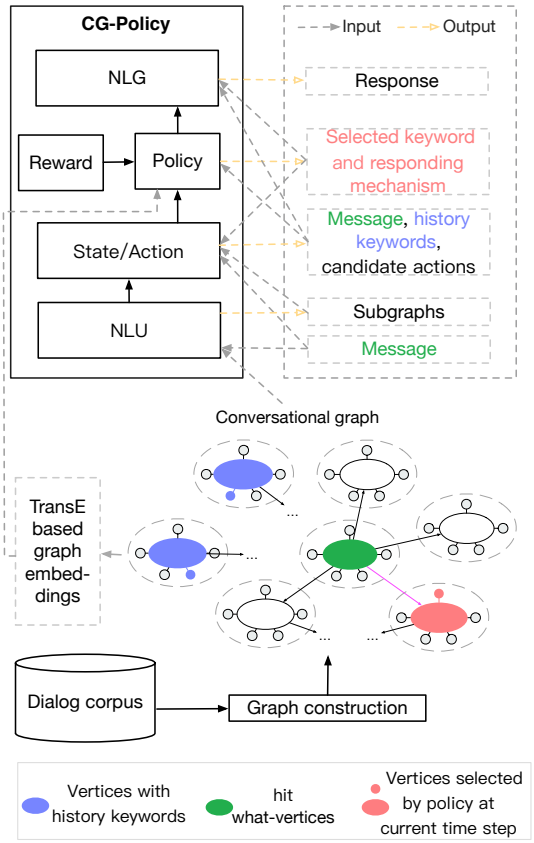

Figure 2: The architecture of our **CG-Policy** that consists of NLU, state/action, policy, and NLG. We first construct conversational graph from dialog corpus. Then we train CG-Policy with RL. The upper-right part shows the details of input/output of each module.

say" modeling. It is reasonable to only adjust the "what-" part when transfer to different domains which further reduces the domain transfer cost.

3 Our Approach

The overview of CG-Policy is presented in Figure 2. Given a user message, to obtain candidate actions, the NLU module attempts to retrieve contextually relevant subgraphs from CG. The state/action module maintains candidate actions, history keywords that selected by policy at previous turns or mentioned by user, and the message. The policy module learns to select a response keyword and a responding mechanism from the above subgraphs. The NLG module first encodes the message into a representation using a message encoder and the selected mechanism, and then employs a Seq2BF model[2] (Mou et al., 2016) to produce a response

[2]It decodes a response starting from the input keyword, and generates the remaining previous and future words subsequently. In this way, the keyword will appear in the response.

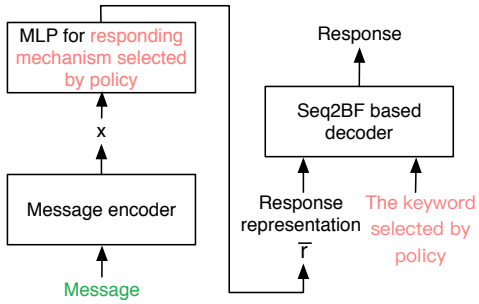

Figure 3: The Multi-mapping based generator for NLG in which we use a Seq2BF based model (Mou et al., 2016) as the decoder.

with the above representation and the selected keyword as input. The models used in CG construction/policy/NLG/reward are trained separately.

3.1 Background: Multi-mapping Generator for NLG

To address the "one-to-many" semantic mapping problem for conversation generation, Chen et al.(2019) proposed an end-to-end multi-mapping model in which each responding mechanism (a MLP network) models how to express response content (e.g. responding with a specific sentence function). In test procedure, they randomly select a mechanism for response generation.

As shown in Figure 3, the generator consists of a RNN based message encoder, a set of responding mechanisms, and a decoder. First, given a dialog message, the message-encoder represents it as a vector $\mathbf{x}$. Second, the generator uses a responding mechanism (selected by policy) to convert $\mathbf{x}$ into a response representation $\bar{r}$. Finally, $\bar{r}$ and a keyword (selected by policy) are fed into the decoder for response generation. To ensure that the given keyword will appear in generated responses, we introduce another Seq2BF based decoder (Mou et al., 2016) to replace the original RNN decoder. Moreover, this generator is trained on a dataset with pairs of [the message, a keyword extracted from a response]-the response.[3]

3.2 CG Construction

Given a dialog corpus D, we construct the CG with three steps: what-vertex construction, how-vertex construction, and edge construction.

[3]If multiple keywords are extracted from the response, we randomly choose one; and if no keyword exists in the response, we randomly sample a word from the response to serve as "keyword".

What-vertex construction To extract content words from D as what-vertices, we use a rule-based keyword extractor to obtain salient keywords from utterances in D.[4] After removing stop words, we obtain all the keywords as what-vertices.

How-vertex construction We obtain a set of N_r responding mechanisms from the generator described in Section 3.1. Then they are used as how-vertices. Notice that all the how-vertices in CG share the same set of responding mechanisms.

Edge construction There are two types of edges in CG. One is to join two what-vertices and the other is to join a what-vertex and a how-vertex. To build the first type of edges, we first construct another dataset that consists of keyword pairs, where each pair consists of any two keywords extracted from the message and the response respectively in D. To capture natural transitions between keywords, we train another multi-mapping based model on this new dataset.[5] For each what-vertex v^w, we find appropriate keywords as its responses by selecting top five keywords decoded (decoding length is 1) by each responding mechanism, and then connect v^w to vertices of these keywords.

To build the second type of edges, for the [message-keyword]-response pair in D (described in Section 3.1), we use the ground-truth response to select the most suitable mechanism for each keyword. Then, given a what-vertex v^w, we select top five mechanisms that are frequently selected for v^w's keyword. Then we build edges to connect v^w to each of the top ranked how-vertices. These edges lead to responding mechanisms that are suitable to generate v^w.

3.3 NLU

To obtain subgraphs to provide high-quality candidate actions, we first extract keywords in the last utterance of the context (message) using the same tool in CG construction, and then link each keyword to the CG through exact string matching, to obtain multiple hit what-vertices. Then we retrieve a subgraph for each keyword, and use vertices (exclude hit what-vertices) in these subgraphs as candidate actions. Each subgraph consists of three parts: the hit what-vertex, its one-hop neighboring what-vertices, and how-vertices being connected to the above neighbors. If there are no keywords to be extracted from the message or to be linked to CG, we reuse the retrieved subgraphs at the last time.[6] Thus we leverage the CG to provide high-quality candidate actions, instead of using the whole set of candidates as done in previous work (Yao et al., 2018).

0. Prepare dataset D and pretrained embedding.
1. Construct the what-vertex set. (3.2)
2. Train a multi-mapping based generator for NLG. (3.1) Responding mechanisms constitute the how-vertex set.
3. Construct edges between two what-vertices or a what-vertex and a how-vertex. (3.2)
4. Train a scoring model for local relevance. (3.6)
5. Train TransE based embedding and PageRank scores for what-vertices. (3.6)
6. Calculate shortest path distances between any two what-vertices. (3.6)
7. Train a original multi-mapping based with a RNN decoder on D for user-simulator. (4.3)
8. Optimize policy with reinforcement learning, where parameters in other modules stay intact. (3.7)

Table 1: The training procedure of CG-Policy.

3.4 State/Action

This module maintains candidate actions, history keywords that selected by the policy or mentioned by user, and the message. Moreover, we use the message-encoder from Section 3.1 to represent the message as a vector $\mathbf{x}$, and then we use all the responding mechanisms from Section 3.1 to convert $\mathbf{x}$ into N_r candidate response representations $\{\mathbf{r}_j\}_{j=1}^{N_r}$, which will be used in the policy.

3.5 Policy

State representation The state representation $\mathbf{s}_t$ at the t-th time step is obtained by concatenating a message representation $\mathbf{s}_t^M$ and a history keywords representation $\mathbf{s}_t^V$ that are encoded by two RNN encoders respectively. Formally,

$$\mathbf{s}_t = [\mathbf{s}_t^M; \mathbf{s}_t^V]. \quad (1)$$

To enable global information aware policy decisions, we employ a graph attention mechanism and graph embedding to encode global structure information into state representation.

Recall that we have a subgraph for each keyword in the message obtained by NLU. Here each subgraph g_i consists of a hit what-vertex,

[4] github.com/squareRoot3/Target-Guided-Conversation

[5] We ever tried other methods for edge construction, e.g., PMI (Yao et al., 2018). Finally we found that our method can provide more diverse response keyword candidates, while PMI tends to provide high-frequency keyword candidates. Here we use a RNN based decoder to replace the Seq2BF.

[6] If we encounter this case at the first time step, hit what-vertices are set as what-vertices that contain the top-5 high-frequency keywords in D.

its what-vertex neighbors (here we remove how-vertices) and edges between them. Formally, $g_i = \{\tau_k\}_{k=1}^{N_{g_i}}$, where each τ_k is a triple with $\tau_k = (head_k, rel_k, tail_k)$, and N_{g_i} is the number of triples in g_i. For non keywords in the message, a NULL subgraph is used.

Then we calculate a subgraph vector $\mathbf{g}_i$ as a weighted sum of head vectors and tail vectors in the triples.

$$\begin{aligned}\mathbf{g}_i &= \sum_{k=1}^{N_{g_i}} \alpha_k [\mathbf{e}_{head_k}; \mathbf{e}_{tail_k}], \\ \alpha_k &= \frac{\exp(\beta_k)}{\sum_{m=1}^{N_{g_i}} \exp(\beta_m)}, \\ \beta_k &= \mathbf{e}_{rel_k}^T \tanh(\mathbf{W}_h \mathbf{e}_{head_k} + \mathbf{W}_t \mathbf{e}_{tail_k}).\end{aligned} \tag{2}$$

Here $\mathbf{e}_*$ represents pretrained graph embedding (TransE (Bordes et al., 2013)) that are not updated during RL training. $\mathbf{W}_h$ and $\mathbf{W}_t$ are parameters.

$\mathbf{s}_t^M$ is obtained by recursively feeding a concatenated vector $\mathbf{e}_i = [\mathbf{w}_i^c; \mathbf{g}_i]$ into a vanilla RNN unit, where $\mathbf{w}_i^c$ (as model parameters) is the embedding of the keyword w_i^c. Thus we encode the global graph structure information into RL state representations, enabling a global-information aware policy model. Moreover, we calculate $\mathbf{s}_t^V$ in a similar way.

Policy decision Each decision consists of two sequential sub-decisions. First the what-policy selects a what-vertex from candidate what-vertices, and then the how-policy selects a how-vertex from how-vertex neighbors of the selected what-vertex.

With $\mathbf{s_t}$ as the state representation, the what-policy μ^{what} is defined by:

$$\mu^{what}(\mathbf{s}_t, \mathbf{v}_j^w) = \frac{\exp(\mathbf{s}_t^T \mathbf{v}_j^w)}{\sum_{l=1}^{N_{w_act}} \exp(\mathbf{s}_t^T \mathbf{v}_l^w)}, \tag{3}$$

where $\mathbf{v}_j^w$ (as model parameters, different from both $\mathbf{w}_i^c$ and $\mathbf{e}_*$) is the embedding of the j-th candidate what-vertices, and N_{w_act} is the number of candidate what-vertices.

The how-policy μ^{how} is defined by:

$$\mu^{how}(\mathbf{s}_t, \mathbf{r}_i) = \frac{\lambda_i \exp(\mathbf{s}_t^T \mathbf{r}_i)}{\sum_{j=1}^{N_r} \lambda_j \exp(\mathbf{s}_t^T \mathbf{r}_j)}, \tag{4}$$

where $\mathbf{r}_i$ is a candidate response representation in the state module, and λ_i is mechanism mask. λ_i is set as 1 if the i-th responding mechanism is one of neighbors of the selected what-vertex, otherwise 0.

3.6 Rewards

Following previous works, we consider these utterance-level rewards:

Local relevance We use a state-of-the-art multi-turn response selection model, DualEncoder in (Lowe et al., 2015), to calculate local relevance.

Repetition Repetition penalty is 1 if the generated response shares more than 60% words with any contextual utterances, otherwise 0.

Target similarity For target-guided conversation, we calculate cosine similarity between the chosen keyword and the target word in pretrained word embedding space as target similarity.[7]

To leverage the global graph structure information of CG to facilitate policy learning, we propose the following rewards:

Global coherence We calculate the average cosine distance between the chosen what-vertex and one of history what-vertices (selected or mentioned previously) in TransE based embedding space (also used in Equation 2) as coherence reward.

Sustainability It is reasonable to promote what-vertices with a large number of neighbors to generate more sustainable, coherent, and diverse dialogs. For this reward, we calculate a PageRank score (calculated on the full CG) for the chosen what-vertex.

Shortest path distance to the target For target-guided conversation, if the chosen what-vertex is closer to the target what-vertex in terms of shortest path distance when compared to the previously chosen what-vertex, then this reward is 1, or 0 if the distance does not change, otherwise -1.

Moreover, we define the final reward as a weighted sum of the above-mentioned factors, where the weight of each factor is set as [0.5, -5, 0, 3, 8000, 0] by default.[8] We see that our rewards can fully leverage dialog transition information in training data by using not only utterance based rewards (e.g., local relevance), but also graph based rewards (e.g., coherence, sustainability).

3.7 Policy Optimization

To make training process more stable, we employ the A2C method (Sutton and Barto, 2018) for optimization. Moreover, we only update policy pa-

[7] If no keyword is chosen, as in baseline models, we calculate target similarity for each word in response and select the closest one.

[8] We optimize these values on Weibo dataset by grid search. The weights of the third/sixth factors are set as 0 by default because they are proposed for target-guided conversation.

rameters, and the parameters of other modules stay intact during RL training.

3.8 NLG

As described in Section 3.1, we use the mechanism selected by how-policy to convert $\mathbf{x}$ into a response representation $\bar{r}$. Then we feed the keyword in the selected what-vertex and $\bar{r}$ into a Seq2BF decoder (Mou et al., 2016) for response generation.

4 Experiments and Results[9]

4.1 Datasets

We conduct experiments on two widely used open-domain dialog corpora.

Weibo corpus (Shang et al., 2015). This is a large micro-blogging corpora. After data cleaning, we obtain 2.6 million pairs for training, 10k pairs for validation and 10k pairs for testing. We use publicly-available lexical analysis tools[10] to obtain POS tag features for this dataset and then we further use this feature to extract keywords from utterances. We use Tencent AI Lab Embedding[11] for embedding initialization in models.

Persona dialog corpus (Zhang et al., 2018a). This ia a crowd-sourced dialog corpora where each participant plays the part of an assigned persona. To evaluate policy controllability brought by CG-Policy, we conduct an experiment for target-guided conversation on the Persona dataset as done in (Tang et al., 2019). The training set / validation set / testing set contain 101,935 / 5,602 / 5,371 utterances respectively. Embeddings are initialized with Glove (Pennington et al., 2014).

Conversational Graph The constructed CG on Weibo corpus contains 4,000 what-vertices and 74,362 edges among what-vertices, where 64% edges are evaluated as suitable for chatting by three human annotators.[12] The constructed CG on Persona corpus contains 1,500 what-vertices and 21,902 edges among what-vertices, where 67% edges are evaluated as suitable for chatting by three human annotators.

4.2 Methods

We carefully select three SOTA methods that focus on dialog policy learning as baselines.

LaRL It is a latent variable driven dialog policy model (Zhao et al., 2019). We use their released codes and choose the multivariate categorical latent variables as RL actions since it performs the best. For target-guided conversation, we implement another model **LaRL-Target**, where we add the "target similarity" factor into RL rewards, and its weight is set as 4 by grid search.

ChatMore We implement the keyword driven policy model (Yao et al., 2018) by following their original design. For target-guided conversation, we implement **ChatMore-Target**, where we add the "target similarity" factor into RL rewards, and its weight is set as 4 by grid search.

TGRM It is a retrieval based model for target-guided conversation, where the keyword chosen at each turn must move strictly closer (in embedding space) to a given target word (Tang et al., 2019). For target-guided conversation, we use the codes released by the original authors, denoted as **TGRM-Target**, and we use their kernel version since it performs the best.[13] To suit the task of open-domain conversation on Weibo, we remove the unnecessary constraint on keyword's similarity with the target word, denoted as **TGRM**.

CG-Policy It is our system presented in Section 3. For target-guided conversation, we implement another system **CG-Policy-Target**, where we use an additional feature, the "shortest path distance to the target" factor, to augment the original what-vertex representation $\mathbf{v}_j^w$ in the what-policy μ^{what}. Formally, $\bar{\mathbf{v}}_j^w = \mathbf{W}_1 * [\mathbf{v}_j^w; \mathbf{e}_{d_j}]$, where $\bar{\mathbf{v}}_j^w$ is the augmented representation, $\mathbf{W}_1$ is a weighting matrix, $\mathbf{e}_{d_j}$ is an embedding for the distance value d_j, and $\bar{\mathbf{v}}_j^w$ has the same size with $\mathbf{v}_j^w$. We also use this factor in reward estimation and its weight is set as 5 by grid search, and we don't use the "target similarity" factor. Moreover, we use the same dialog corpora to construct CG, train user simulator, reward functions, and the NLG module for CG-Policy.

4.3 User Simulator

We use the same user simulator for RL training of LaRL, ChatMore and CG-Policy. The user simulator is the original multi-mapping based generator with a RNN decoder, which is pretrained on dialog corpus and not updated during policy training. Please refer to (Chen et al., 2019) for more details. During testing, all the systems share this simulator.

[9] Please see the supplemental material for more details.
[10] ai.baidu.com/
[11] ai.tencent.com/ailab/nlp/embedding.html
[12] We randomly sample 500 edges for evaluation.
[13] github.com/squareRoot3/Target-Guided-Conversation

4.4 Evaluation Settings

Conversation with user simulator Following previous work (Li et al., 2016b; Tang et al., 2019), we use a user simulator to play the role of human and let each of the models converse with it. Given a randomly selected model, we randomly select an utterance from all the utterances (at the starting position of sessions) in test set for the model to start a conversation. Moreover, we set a maximum allowed number of turns, which is 8 in our experiment. Finally, we collect 100 model-simulator dialogs for evaluation. For single-turn level evaluation, we randomly sample 100 message-response pairs from the dialogs for each model.

Conversation with human Following previous work (Tang et al., 2019), we also perform human evaluation for a more reliable system comparison. Given a model to be evaluated, we randomly select a dialogue from test set and pick its first utterance for the model to start a conversation with a human. Then the conversation will continue till 8 turns are reached. Finally, we obtain 50 dialogs for evaluation. For single-turn level evaluation, we randomly sample 100 message-response pairs from the dialogs for each model.

4.5 Evaluation Metrics

Metrics such as BLEU and perplexity have been widely used for dialog evaluation (Li et al., 2016a; Serban et al., 2016), but it is widely debated how well these automatic metrics are correlated with true response quality (Liu et al., 2016). Since the proposed system does not aim at predicting the highest-probability response at each turn, but rather the long-term success of a dialog (e.g., coherence), we do not employ BLEU or perplexity for evaluation, and we propose the following metrics.

4.5.1 Multi-turn Level Metrics

Global coherence We define incoherence problems as follows: (1) Inconsistent dialogs where the model contradicts with itself, e.g., the model says he is a driver before and then says he is a doctor; (2) One-side dialogs in which the model ignores the user's topics with two or more consecutive turns. A session will be rated "0" if it contains more than three incoherence cases, or "+1" if a session contains 2 or 3 cases, otherwise "+2".

Distinct The metric Dist-i calculates the ratio of distinct i-gram in generated responses (Li et al., 2016a). We use Dist-2 to measure the diversity of generated responses.

Methods	Cohe.	Dist-2	Appr.	Infor.
LaRL	0.85	0.12	0.55	0.77
ChatMore	0.95	0.05	0.58	0.93
TGRM	0.79	**0.42**	0.68	1.00
CG-Policy	**1.33**	0.31	**0.73**	1.00

Table 2: Results for dialogs with simulator on Weibo.

Dialog-target success rate For target-guided conversation, we measure the success rate of generating the target word within 8 turns.

4.5.2 Single-turn Level Metrics

Local appropriateness[14] A response will be rated "0" if it is inappropriate as an reply to the given message, otherwise "1".

Informativeness "0" if a response is a "safe" response, e.g. "I don't know", otherwise "1".

4.6 Evaluation Results

4.6.1 Setting

We ask three annotators to judge the quality of each dialog (at multi-turn level) or utterance pair (at single-turn level) for each model. Notice that model identifiers are masked during evaluation.

4.6.2 Conversation with simulator

As shown in Table 2, CG-Policy significantly outperforms (sign test, **p-value** $< \mathbf{0.01}$) baselines in terms of global coherence and local appropriateness. It indicates that the CG can effectively facilitate policy learning (see the ablation study for further analysis). For LaRL, its single-turn response quality is worse than other models. It might be explained by that their latent variables are not fine-grained enough to provide sufficient information to guide response generation. ChatMore tends to select high-frequency or generic keywords, resulting in its worst performance in terms of Dist-2. TGRM performs the best in terms of Dist-2 and informativeness, indicating that retrieval-based models can produce more diverse responses than generation based models. It is consistent with the conclusions in previous work (Chen et al., 2017; Zhang et al., 2018a). However, TGRM performs the worst in terms of coherence, since TGRM does not use RL framework. It indicates the importance of RL framework for multi-turn dialog modeling. Here the Kappa value for inter-annotater agreement is above 0.4, indicating moderate agreement.

[14] We do not consider if a response is appropriate or not for the selected responding mechanism.

Methods	Cohe.	Dist-2	Appr.	Infor.
LaRL	0.82	0.22	0.52	0.74
ChatMore	0.88	0.15	0.54	0.94
TGRM	0.77	**0.63**	0.61	1.00
CG-Policy	**1.26**	0.47	**0.67**	1.00

Table 3: Results for dialogs with human on Weibo.

Methods	Cohe.	Dist-2	Appr.	Infor.
CG-Policy	**1.33**	**0.31**	**0.73**	1.00
ChatMore	0.95	0.05	0.58	0.93
ChatMore-CG	1.15	0.14	0.65	0.91
CG-Policy-noCG	1.03	0.07	0.62	1.00
CG-Policy-noCGact	1.11	0.08	0.68	1.00
CG-Policy-noCGrwd	1.06	0.19	0.64	1.00
CG-Policy-noCGhow	1.21	0.13	0.65	1.00

Table 4: Ablation study for CG-Policy on Weibo.

4.6.3 Conversation with human

As shown in Table 3, CG-Policy outperforms baselines in terms of both global coherence and local appropriateness (sign test, **p-value** < 0.01) , which is consistent with the results in Table 2. The Kappa value is above 0.4, indicating moderate agreement.

4.6.4 Ablation study

We conduct an ablation study for CG-Policy on Weibo corpus to investigate why CG-Policy performs better. **First**, to evaluate the contribution of CG, we remove the CG from CG-Policy, denoted as CG-Policy-noCG, where we do not use graph structure information for action space pruning and reward design. Moreover, we attempt to use the CG (without how-vertices) to augment the ChatMore model for action space pruning and reward design, denoted as Chatmore-CG. As shown in Table 4, the performance of CG-Policy-noCG drops dramatically in terms of coherence, Dist-2 and appropriateness when compared to the original model. Moreover, CG can boost the performance of ChatMore in terms of most of metrics. It indicates that the use of CG is crucial to the superior performance of CG-Policy, and it can also help other models, e.g., ChatMore. **Second**, to evaluate the contribution of CG for action space pruning or reward design respectively, we implement two system variants: (1) we use all the what-vertices in CG as action candidates at each turn, denoted as CG-Policy-noCGact; (2) we remove all the CG-based factors from RL rewards, denoted as CG-Policy-noCGrwd. As shown in Table 4, the performance of CG-Policy-noCGact drops significantly in terms of Dist-2 as it tends to select high-frequency keywords like ChatMore, indicating the importance of graph paths to provide both locally-appropriate and diverse response keywords. Moreover, the performance of CG-Policy-noCGrwd drops significantly in terms of coherence, indicating that CG based rewards can effectively guide CG-Policy to promote coherent dialogs. **Third**, we remove how-vertices from CG, denoted as CG-Policy-noCGhow. As shown in Table 4, how-vertex removal hurts its performance in Dist-2, indicating the importance of how-vertices for response diversity.

4.7 The Task of Target-guided Conversation

Besides maintaining coherence, CG grounded policy learning can enable more control over dialog models, which is important to achieve certain goals for chatbot, e.g. proactive leading to certain chatting topics (keywords) or certain products.

4.7.1 Setting

Following the setting in (Tang et al., 2019), where we randomly sample a keyword as the target word for each session in testing procedure. Here we use a multi-mapping based user simulator trained on the Persona dataset for evaluation.

Methods	Succ.(%)	Cohe.	Appr.	Infor.
LaRL-Target	1	0.91	0.62	0.91
ChatMore-Target	6	0.93	0.65	0.97
TGRM-Target	69	0.96	0.67	1.00
CG-Policy-Target	**98**	**1.17**	**0.75**	1.00

Table 5: Results for target-guided dialogs on Persona.

4.7.2 Results

Table 5 presents the results on 100 dialogs for each model. We see that CG-Policy-Target can significantly outperform baselines in terms of dialog-target success rate (sign test, **p-value** < 0.01). It can be seen that that CG-Policy can successfully lead the dialog to a given target word by learning to walk over the CG, indicating that this graph gives us more control over the policy. LaRL-Target and ChatMore-Target perform badly in terms of success rate. It may be explained by that they lack the ability of proactive dialog content planning.

4.8 Analysis of Responding Mechanisms

Figure 4 provides representative words of each mechanism.[15] For example, for Mech-1, its keywords are mainly subjective words (e.g. think) for

[15] We select words that occur frequently in responses guided by this mechanism but rarely occur with other mechanisms.

generation of responses with respect to personal opinion or intention. For Mech-2, it tends to respond with a specific type of mood.

Mech-1	Mech-2	Mech-3	Mech-4	Mech-5
以为 think	哈哈 haha	哪 where	漂亮 beautiful	别 no
想 want	哇 wow	什么 what	可爱 cute	还是 or else
信 believe	好吧 alright	？	萌 cuddly	没有 no

Figure 4: Representative words of responding mechanisms.

5 Conclusion

In this paper we present a novel graph grounded policy learning framework for open-domain multi-turn conversation, which can effectively leverage prior information about dialog transitions to foster a more coherent and controllable dialog. Experimental results demonstrate the effectiveness of this framework in terms of local appropriateness, global coherence and dialog-target success rate. In the future, we will investigate how to extend the CG to support hierarchical topic management in conversational systems.

Acknowledgments

We are grateful for the support from Yan Zeng at the initial stage of this work. We also thank the anonymous reviewers for their helpful comments and suggestions. This work is supported by the National Key Research and Development Project of China (No.2018AAA0101900) and the National Natural Science Foundation of China (NSFC) via grant 61976072.

References

Siqi Bao, Huang He, Fan Wang, Rongzhong Lian, and Hua Wu. 2019. Know more about each other: Evolving dialogue strategy via compound assessment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5382–5391.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Chaotao Chen, Jinhua Peng, Fan Wang, Jun Xu, and Hua Wu. 2019. Generating multiple diverse responses with multi-mapping and posterior mapping selection. *Proceedings of IJCAI*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *SIGKDD Explorations*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *Proceedings of ICLR*. Association for Computational Linguistics.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of AAAI 2018*, pages 5110–5117. Association for the Advancement of Artificial Intelligence.

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2333–2343.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453.

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

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. 2016b. Deep reinforcement learning for dialogue generation. In *Proceedings of EMNLP*, pages 1192—-1202.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019. Knowledge aware conversation generation with explainable reasoning over augmented graphs. In *EMNLP-IJCNLP*.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. 2018. Towards exploiting background knowledge for building conversation systems. In *Proceedings of EMNLP*, pages 2322—-2332. Association for Computational Linguistics.

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 COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3349–3358.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of AAAI*, pages 3776—-3784.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of ACL-IJCNLP*, volume 1, pages 1577–1586.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT Press.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric P. Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of ACL*.

Jun Xu, Haifeng Wang, Zhengyu Niu, Hua Wu, and Wanxiang Che. 2020. Knowledge graph grounded goal planning for open-domain conversation generation. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Lili Yao, Ruijian Xu, Chao Li, Dongyan Zhao, and Rui Yan. 2018. Chat more if you like: Dynamic cue words planning to flow longer conversations. *arXiv preprint arXiv:1811.07631*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018a. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Wei-Nan Zhang, Lingzhi Li, Dongyan Cao, and Ting Liu. 2018b. Exploring implicit feedback for open domain conversation generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1208–1218.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of IJCAI-ECAI*.

A Appendices

Training Details and Two Conversation Cases For fair comparison, all models share the same vocab (maximum size is 50000 for the Weibo corpus or 20000 for the Persona corpus), initialized word embedding (size is 200), and keyword set. Further, One layer bidirectional GRU-RNN (hidden size is 512) is utilized for all encoders. Dropout rate is 0.3, and optimizer is Adam(lr=2le-3) for all models.

We initialize each session with a starting utterance chosen from the training dataset randomly. Maximum turns is set as 8 and the discounting weight for rewards is set as 0.95.

LaRL	
Categorical z	M=10,K=20
β	0.01
CG-Policy	
TransE Embedding Size	100
Number of Responding Mechanisms N_r	10

Table 6: Training details for models.

B：	晚饭吃了么 [Did you have dinner?]
U：	我想吃排骨了，晚上炖排骨吃吧 [I want to have spareribs. Let's stew spareribs in the evening.]
B：	我请你吃猪蹄吧，我也要吃 [I'll treat you to pig hooves. I'll eat them, too.]
U：	炸猪蹄很好吃啊 [Fried pig hooves are delicious.]
B：	这是什么类型啊，你吃过么？ [What kind of food is it? Have you eaten it?]
U：	我还是吃炸酱面吧，最近减肥 [I'd better eat fried sauce noodles. I've lost weight recently.]
B：	正宗了，我也想吃 [It's authentic. I want to eat it, too.]
U：	我想跟你一起跳舞 [I want to dance with you.]
B：	好萌啊，好可爱，好喜欢 [how budding! You are lovely and I like it.]
U：	我想起我家那条狗了，泰迪狗，给你看看照片 [I think of my dog, Teddy. Show you the picture.]
B：	我想养狗了，我也要养一只 [I want to have a dog. I want to have one, too.]
U：	我想回家吃狗肉 [I want to go home and eat dog meat.]
B：	可怜的泰迪啊，哈哈哈 [Poor Teddy, haha]
U：	我不是要回家吃我的泰迪！[I'm not going home to eat my Teddy!]

Figure 5: Case 1: One conversation between CG-Policy and human, where “B” is CG-Policy and “U” is human. The red words are keywords. We translate the original Chinese utterances into English.

B：	昨天去菜市场买了一大袋土豆，累死我了 [Yesterday I went to the vegetable market and bought a big bag of potatoes. I am exhausted.]
U：	话说楼下新开了个超市 [I heard that there is a new supermarket downstairs.]
B：	我去了， 不打折了，哈哈 [I've been there. No discount yet, haha.]
U：	里面的水果新鲜么 [Is the fruit fresh?]
B：	我都没看出是有多难吃 [It’s untasty, even more than I expected.]
U：	刚开业就不打折,真抠门,我去淘宝买吧 [It’s really stingy that have no discount just after opening. I'll go to Taobao to buy fruit.]
B：	在哪里买啊，低价代购？求地址 [Which shop? Substitute purchase at low price? Give me the link address.]
U：	很多店铺吧, 搜一下就好了 [A lot of shops can be your choice. Just search them.]
B：	万能的淘宝网店，天猫要火啊 [Taobao shops are really omnipotent, TianMao is becoming more and more famous]
U：	双十一要到了，马云赚钱好开心 [The Double Eleventh Shopping Festival is coming. Ma Yun is happy to earn lots of money.]
B：	我去年双十一购买了5件商品 [I bought five items last year in the Double Eleventh Shopping Festival.]
U：	美死了,哈哈,好可爱,哈哈 [Haha, you are lovely, haha]
B：	你也很可爱 [You're lovely, too]
U：	我想买一只萨摩耶，好可爱 [I want to buy a Samoa. It's cute.]

Figure 6: Case 2: One conversation between CG-Policy and human, where “B” is CG-Policy and “U” is human. The red words are keywords. We translate the original Chinese utterances into English.