Towards Large-Scale Interpretable Knowledge Graph Reasoning for Dialogue Systems

Yi-Lin Tuan¹, Sajjad Beygi², Maryam Fazel-Zarandi², Qiaozi Gao², Alessandra Cervone², William Yang Wang¹
¹ University of California, Santa Barbara  ² Amazon Alexa AI
{ytuan, william}@cs.ucsb.edu
{beygi, fazelzar, qzgao, cervon}@amazon.com

Abstract
Users interacting with voice assistants today need to phrase their requests in a very specific manner to elicit an appropriate response. This limits the user experience, and is partly due to the lack of reasoning capabilities of dialogue platforms and the hand-crafted rules that require extensive labor. One possible way to improve user experience and relieve the manual efforts of designers is to build an end-to-end dialogue system that can do reasoning itself while perceiving user’s utterances. In this work, we propose a novel method to incorporate the knowledge reasoning capability into dialogue systems in a more scalable and generalizable manner. Our proposed method allows a single transformer model to directly walk on a large-scale knowledge graph to generate responses. To the best of our knowledge, this is the first work to have transformer models generate responses by reasoning over differentiable knowledge graphs. We investigate the reasoning abilities of the proposed method on both task-oriented and domain-specific chit-chat dialogues. Empirical results show that this method can effectively and efficiently incorporate a knowledge graph into a dialogue system with fully-interpretable reasoning paths.

1 Introduction
Nowadays, dialogue systems are ubiquitous in customer service and voice-based assistants. One of the main uses of this technology is supporting humans in accomplishing tasks that might require accessing and navigating large knowledge bases (e.g., movies search). A dialogue system architecture is typically composed of a natural language understanding (NLU) module, a dialogue management (DM) module, and a natural language generation (NLG) module (Jurafsky and Martin, 2009; Williams et al., 2016). First, the NLU component extracts a meaning representation from the user utterance based on which the DM generates the next system action by reasoning over the meaning representation and communicating with external applications if necessary. For example, the DM may retrieve information from external knowledge graphs (KG) to answer the user’s query based on the dialogue history. This process requires the DM to convert the output of NLU to a query to be issued to the backend. Given the difficulty of this step, which is often domain-dependent, the DM component might require the design of hand-crafted rules. However, such rules are usually not scalable to different applications. They could require considerable effort to cover all possible cases/dialogue flows, leading to expensive costs to design new applications. Moreover, in several cases, users interacting with such assistants are forced to formulate specific queries in order to accomplish their objective, which might break user engagement.

To alleviate the problem of having to design expensive hand-crafted rules and breaking user experience, recent works have explored the possibility of building end-to-end dialogue systems (Wen et al., 2017) and all-in-one response generation models (Serban et al., 2016). Among them, since graph is one of the main structure to store knowledge, recent research (Ghazvininejad et al., 2018; Zhou et al., 2018; Moon et al., 2019; Tuan et al., 2019; Yang et al., 2020) has proposed methods to generate natural language responses according to both the dialogue history and external knowledge graph. Despite these innovative and inspiring methods, there are some shortcomings. For instance, these methods are either not fully-interpretable or limited to small-scale knowledge graphs.

In this paper, we propose a novel dialogue differentiable knowledge graph model (DiffKG). The DiffKG is a single transformer model that directly (1) generates a sequence of relations to perform multi-hop reasoning on a reified KG representation proposed by (Cohen et al., 2019), and then (2) generates responses using the retrieved entities. To the best of our knowledge, this is the first dialogue
model that can directly walk on a large-scale KG with flexibility and interpretability. DiffKG allows having flexible entity values in the KG and handling novel entity values with an arbitrarily defined number of tokens. The reasoning path of DiffKG consists of the predicted relations, thus allowing for transparency.

We run extensive experiments to test DiffKG performance on KG-grounded dialogues. We select Stanford Multi-domain Dialogues (SMD) (Eric et al., 2017) and propose a new dataset, SMD-Reasoning, to simulate scenarios requiring multiple reasoning types and select the OpenDialKG (Moon et al., 2019) to simulate scenarios requiring large-scale KG reasoning without preprocessing. We then compare DiffKG with state-of-the-art models on SMD and OpenDialKG and an additional baseline that flattens KGs into a textual form from which transformers can learn. Empirically, our experiments show that DiffKG can effectively be trained on large-scale KGs and demonstrate its robustness with modified triplets in a KG. From the perspective of computation, DiffKG leads to relatively low extra time and memory usage compared to transformer models not using any KG information.

In summary, our contributions are: 1) We propose DiffKG, a novel method that can effectively and flexibly incorporate large-scale KG; 2) We demonstrate that DiffKG is a model-agnostic method and can be applied to different model architectures; 3) We show that DiffKG is an interpretable method with low add-on latency at inference time. Our code and processed datasets are released in https://github.com/Pascalson/DiffKG-Dialog.

2 Related Work

Recent years have seen a surge of new methods proposing end-to-end models that try to both understand natural language input text and search information. Two of the widely explored tasks are question-answering (QA) and dialogue generation.

QA. Multiple QA methods (Weston et al., 2015; Yin et al., 2016; Hao et al., 2017; Rajpurkar et al., 2018; Verga et al., 2020; Eisenschlos et al., 2021) have been proposed to tackle tasks that go beyond what is explicitly stated in the linguistic context (Storks et al., 2019). For example, the benchmarks (Mihaylov et al., 2018; Reddy et al., 2019; Khot et al., 2020; Lin et al., 2021) are particularly useful for the model to extract information from external knowledge bases to answer questions. Nonetheless, these studies mostly take the retrieved information from KG as the answer to a single question, while in dialogue we have to formulate an informative response to multi-turn dialogue history.

Dialogue Generation. Recent works have investigated the grounded dialogue generation. These methods can be divided into three main categories. First, Dinan et al. (2018); Zhao et al. (2019); Tuan et al. (2020); Kim et al. (2020) extract useful knowledge from unstructured data to generate responses, such as information contained in passages and speaker’s profiles. Second, Sordoni et al. (2015); Long et al. (2017); Zhu et al. (2017); Ghazvininejad et al. (2018); Zhou et al. (2018); Veličković et al. (2018); Joshi et al. (2020); Hosseini-Asl et al. (2020); Wang et al. (2021) utilize information from knowledge bases (either graphs or tables) to enhance the dialogue system. They usually train the entities and relations embeddings of the knowledge bases and incorporate these embeddings into the input representation to predict the response. Third, Moon et al. (2019); Tuan et al. (2019); Jung et al. (2020) formulate the reasoning process more explicitly, as a path traversal over knowledge graphs. These methods further improve the transparency and explainability of the conversational agent and share the most similar idea with us. However, they either only predict the reasoning path without generating responses or need subgraph sampling to reduce the scale of KG. In this work, our approach uses a transformer model to jointly predict explicit reasoning paths over a large-scale knowledge graph and generate dialogue responses based on the reasoning results.

3 Background

3.1 Knowledge Graph for Dialogue System

We assume that the knowledge of the system can be represented by a knowledge graph (KG) \( \mathcal{G} = \{ \mathcal{E}, \mathcal{R} \} \), where \( \mathcal{E} \) denotes the entities and \( \mathcal{R} \) denotes the relations. The knowledge graph \( \mathcal{G} \) contains multiple triples describing the connections among entities and relations. We denote the \( k \)-th triple of this graph as \((e^h_k, r_k, e^t_k)\), where \(e^h_k, r_k, e^t_k\) are respectively the head entity, relation, and tail entity.

The total numbers of triples, entities, and relations
Table 1: Example of different reasoning types and output formats (semantic and natural language forms) in a dialogue system with the related information in the accessible KGs.

<table>
<thead>
<tr>
<th>Reasoning Type</th>
<th>Example</th>
<th>Related Info. in KG</th>
</tr>
</thead>
<tbody>
<tr>
<td>KG reasoning</td>
<td>U: I need unleaded gas. R: inform Valero, 4 miles</td>
<td><img src="https://via.placeholder.com/150" alt="Image" /></td>
</tr>
<tr>
<td>Logical Reasoning</td>
<td>True/False</td>
<td>U: Is it going to snow this week at Corona? R: Yes</td>
</tr>
<tr>
<td>Selection</td>
<td>U: give me the direction to the nearest shopping mall. R: inform Stanford Shopping Center, 3 miles</td>
<td><img src="https://via.placeholder.com/150" alt="Image" /></td>
</tr>
<tr>
<td>Extraction</td>
<td>U: What gas stations are here? R: include poi_type gas station</td>
<td><img src="https://via.placeholder.com/150" alt="Image" /></td>
</tr>
</tbody>
</table>

are denoted as $N_T$, $N_E$, $N_R$, respectively.\(^1\)

3.2 Response Generation in Dialogue System

If we define the dialogue history as a sequence of tokens that occurred during the user and system interactions, then a flattened dialogue history can be written as:

$$x = (x_1, x_2, ..., x_m, ..., x_M)$$  \hspace{1cm} (1)

where $x_m$ is the $m$-th token in the dialogue history with $M$ tokens. In an end-to-end dialogue system, we assume a dialogue system parameterized by $\theta$ exists that can predict a probability distribution of responses $P(\cdot | x, \mathcal{G})$. The generated responses are sampled from this probability distribution.

4 Problem Statement

We focus on understanding the ability of language models in performing reasoning during a conversation. We consider two tasks that are usually required in dialogue scenarios and call them semantic form and natural language (NL) form in Table 1. First, given a dialogue history and a user’s query, the task of semantic form is to predict the next system action, corresponding to the output of the DM module, based on the available knowledge. In this case, we assume the expected output is the essential knowledge for an NLG module. We argue that this task can help better evaluate if the response is correct or not and which type of reasoning can be more successfully handled. Second, given a dialogue history and a user’s query, the task of the NL form could be to directly output the response given by the system. This setting with annotated reasoning path can shed light on understanding if the model can learn to support chit-chat and reasoning at the same time.

Moreover, we aim to understand models’ reasoning capability both in the form of logical reasoning and over the provided knowledge. As illustrated in Table 1, by KG reasoning, we refer to the ability of the model to retrieve information from an arbitrary scaled KG in multiple hops. Meanwhile, we refer to logical reasoning as the ability of the model to conduct operations such as evaluating whether a statement is true or false, selecting min/max from a list of alternatives, and extracting constraints.

We formulate the task that we focus on as follows: given the dialogue history $x$ and currently accessible KG $\mathcal{G}$, can we extend a transformer model to predict a correct response $y$ in either semantic or NL form? As illustrated in Table 1, this task not only requires the model to accurately retrieve information from the KG, but also needs to do further logical operations on the information. To solve this task, a model should also be able to effectively integrate the dialogue history $x$ with the KG $\mathcal{G}$.

5 Proposed Approach

Figure 1 illustrates our proposed architecture which contains four main parts: a dialogue history encoder, a differentiable KG reasoning module, a learnable logical operations module, and a response decoder (the transformer model). Note that we experiment with two types of transformers: a causal language model GPT2 (Radford et al., 2019) and an encoder-decoder model T5 (Raffel et al., 2020). For GPT2, we reuse the same encoder that is used at the beginning of the process, i.e., $f_{enc}$ in Figure 1, as the final transformer that generates the response.
We use encoder model to project \( x \) walking module and the operation vector \( c \) where \( H \) walking block and \( c \) operation layer predicts the operation vector \( f \) at the \( h \)th hop to determine the weights of the programmed layer produces the concatenation of a sequence \( W \) points layer with parameters \( \tilde{w} \). At the same time, the embedding \( \tilde{x} \) is also fed into a relation layer with parameters \( W_r \). An entry \((i, r)\) in matrix \( M_h \) or \( M_r \) with value 1 indicates that the \( i \)-th triple in the KG has entity \( e \) as the head or the tail; an entry \((i, r)\) in matrix \( M_t \) with value 1 indicates that the \( i \)-th triple in the knowledge graph has the relation \( r \). Since often in practical settings most entries in the three matrices are zero, saving them into sparse matrices can significantly reduce memory consumption (Cohen et al., 2019).

To ensure that our model can scale to larger KGs, we adopt the reified KG representation proposed by (Cohen et al., 2019). The reified KG represents the graph \( G \) using three sparse matrices: head matrix \( M_h \in \mathbb{R}^{N_T \times N_E} \), relation matrix \( M_r \in \mathbb{R}^{N_T \times N_E} \), and tail matrix \( M_t \in \mathbb{R}^{N_T \times N_E} \). An entry \((i, e)\) in matrix \( M_h \) or \( M_r \) with value 1 indicates that the \( i \)-th triple in the KG has entity \( e \) as the head or the tail; an entry \((i, r)\) in matrix \( M_t \) with value 1 indicates that the \( i \)-th triple in the knowledge graph has the relation \( r \). After predicting the relation sequence \( r \), we start the graph traversal from a given set of initial entities \( E_0 \subseteq E \). We first map the initial entities into a vector \( e_1 = [\mathbb{1}(e \in E_0), \forall e \in E] \). That is, each entry of \( e_1 \in \mathbb{R}^{|E|} \) has value 1 if that entity is in the initial entities list \( E_0 \), otherwise, the entry is zero. We then predict the next (temporary) entity vector \( e_2 \) by conducting a \texttt{Next} module:

\[
e_{h+1} = \text{Next}(e_h, r_h),
\]

where

\[
\text{Next}(e_h, r_h) = \frac{M^T_h (M_h e_h \odot M_r r_h)}{||M_h (M_h e_h \odot M_r r_h)||_2 + \epsilon}.
\]

Here \( \epsilon \) is an arbitrary small number to offset the denominator and prevent division by zero. We introduce the normalized \texttt{Next} to solve the issue with the method proposed by (Cohen et al.,

![Diagram](image.png)

Figure 1: The illustration of proposed DiffKG, which leverages a pretrained transformer model (T5 or GPT2) and the Reified KG. The model generates the response depending on the predicted relation sequence \([r_1; \ldots; r_H]\), thus being fully interpretable in terms of the used reasoning path.
for knowledge graph completion defined as 
\[
\text{Follow}(e_h, r_h) = M^T_h (M_h e_h \odot M_r r_h);
\]
since in a dialogue model, we can seldom predict the relation vectors that perfectly match the entity vectors. That is, if directly using the `Follow` module in (Cohen et al., 2019), the \(|e_h|_2\) will not be one and will vanish as the hop number \(h\) increases. Specifically, note that in our proposed module, the predicted relations \(r_h\) are independent of the traversed entities \(e_h\). For instance, finding the “distance” of “the nearby gas station” is independent of whether the nearby gas station is “Chevron” or “Shell”.

To allow the model to dynamically select the number of reasoning hops, we add a relation type “ToSelf” into \(R\) and connect each entity to itself by “ToSelf”. More specifically, the KG will contain triples \((e_k^h, r_k, e_k^h)\) for all \(e_k^h = e_k^h \in E\) and \(r_k = \text{ToSelf}\).

5.3 Entity Embeddings

At each hop, we further conduct the operation vectors \(a\) on the entities weighted by the entity vector \(e_h\). First, we tokenize each entity and represent it by the concatenation of its token embeddings. This step allows (1) representing entities with longer texts such as phrases and sentences, and (2) eliminating the effort to retrain entity embeddings whenever new entity values are added. The entity embeddings can then be represented as a tensor \(E \in \mathbb{R}^{N_E \times d \times m}\), where \(m\) is the maximum number of tokens of entities\(^2\).

5.4 Learnable Logical Operation and Checkpoints

We compute the transformed entity embeddings by element-wise multiplication of the entity embeddings \(E\) with the entity vector \(e_h\) at the \(h\)-th hop. Next, the dot product of the operation vectors and the transformed entity embeddings is passed to a softmax layer as the entity vector at the next hop:

\[
e_{h+1}^a = \text{softmax}(a(E \odot e_h)), \quad (5)
\]

Further, at the \(h\)-th hop we use the walk-or-check vector \(c_h\) to combine the `Next` and operation modules above. The combined entity vector is given by:

\[
e_{h+1}^a = c_h^T \begin{bmatrix} e_{h+1}^a \\ e_{h+1}^e \end{bmatrix} = c_h^T \begin{bmatrix} \text{Next}(e_h, r_h) \\ \text{softmax}(a(E \odot e_h)) \end{bmatrix}, \quad (6)
\]

5.5 Response Decoder

After \(H\) hops reasoning is done, the entities with top-\(k\) values in the entity vector \(e_H\) are selected, indicating that they have the highest probability to be retrieved from the graph. These entities are converted into their embeddings in \(E\) and multiplied by their values in \(e_H\). These entity embeddings are then concatenated with the dialogue history \(x\). The concatenated vectors are fed as the input into the transformer model to predict the response token by token. The predicted probability distribution over the output space can be written as \(P(x | x, M_h, M_r, M_t)\). Since all components are differentiable, all modules can be trained end-to-end with the dialogue history \(x\) and the reified KG representation \(\{M_h, M_r, M_t\}\) using the cross-entropy loss with the ground-truth output \(y\) as the labels.

\[
L = \sum_{(x,y)} - \log P(y | x, M_h, M_r, M_t). \quad (7)
\]

During the inference time, the reasoning modules (relation layer, operation layer, and checkpoints layer) work exactly the same as the training stage, the only difference is that the response decoder is fed with predicted tokens in previous time steps (inference stage) instead of the ground-truth output (training stage).

6 Experiments

6.1 Datasets

We evaluate our proposed approach on three datasets. Among them, we use Stanford Multi-domain Dialogues (SMD) (Eric et al., 2017) and OpenDialKG (Moon et al., 2019) to test the methods generalizability on different dialogue types (task-oriented / chit-chat) and scales of structured knowledge (pairwise database / universal KG). To further analyze the reasoning ability, we propose a new dataset, SMD-Reasoning, by modifying the output of SMD dataset from natural language responses to actions paired with their reasoning types.

\(^2\)In our experiments, we compute the maximum length of all entities and pad shorter entities to the length of \(m\).
**Stanford Multi-domain Dialogues (SMD)** The SMD dataset (Eric et al., 2017) is composed of two-speaker conversations, where a driver talks with the car assistant to tackle tasks in three domains: scheduling, navigation, and weather forecasting. Each dialogue focuses on one domain and is paired with a database having the related information. We convert the original database into two formats: (1) the natural language descriptions (NLD) and (2) the KG. The NLD form allows us to investigate the ability of the model to interpret unstructured knowledge, while the KG form could be a more extensible structured knowledge compared to tables.

**OpenDialKG** OpenDialKG dataset (Moon et al., 2019) is composed of two-speakers recommendation and chit-chat style conversations. Each turn in a dialogue is annotated with the reasoning path on the provided KG, which is filtered from Freebase (Bollacker et al., 2008). The resulting KG has 1,190,658 triples, 100,813 entities and 1,358 relations. We randomly split 70/15/15% for train/valid/test sets as described in (Moon et al., 2019; Jung et al., 2020) since they do not release their splits.

**SMD-Reasoning** To make SMD dataset suitable for more precise evaluation of reasoning abilities, we manually label and convert it to the SMD-Reasoning dataset. We first remove the natural language part from the original responses and only leave the action word (e.g., inform) along with the information being conveyed. We divide the dataset into three main reasoning types: informing items, selecting min/max, and evaluating true/false. To validate if the models can identify whether the needed knowledge is in the database, we add a new reasoning type for extracting constraints, by removing the needed knowledge from the database and changing the output to “include [knowledge description]” as shown in Table 1. See Appendix A,B for statistics of these datasets.

**6.2 Evaluation Metrics** We use different evaluation methods for the three datasets. For SMD, we follow prior work (Yang et al., 2020) and use BLEU (Papineni et al., 2002), and Entity F1 scores on each domain. For OpenDialKG, we follow the descriptions in prior works (Moon et al., 2019; Jung et al., 2020) to evaluate the path@k scores, i.e., if the ground-truth path is ranked top-k in the predicted paths probabilities. Moreover, since our method not only can predict the reasoning path as prior works but also can predict the response, we also use the BLEU score to get the approximated evaluation of the response quality compared to ground-truth. Note that prior work has discussed that BLEU scores may not match human intuition (Liu et al., 2016), but we use them here as an approximated evaluation for reference.

For SMD-Reasoning, the output is more deterministic and does not include diverse sentence structures. Therefore, we compute the F1 score and the exact match (EM) score of prediction and the ground-truth. The EM score is calculated by removing the order of the prediction since the labels of SMD-Reasoning dataset follow the order of knowledge description appearing in the original ground-truth responses and may not have the same order as generated outputs. The EM score can be written as:

\[
EM = \frac{1}{T} \sum \mathbb{1}(\text{sort}(\hat{y}) = \text{sort}(y)).
\] (8)

where \(\hat{y}\) is inferred from the model using argmax sampling and \(T\) is total number of examples.

**6.3 Implementation Details** Since the proposed method is model-agnostic, we implement it on GPT2 (Radford et al., 2019) and T5 (Raffel et al., 2020). Specifically for the T5 model, we use the unifiedQA-T5 model (Khashabi et al., 2020) which is pretrained on question answering tasks that also need to do reasoning. However, we empirically find that T5 generally has better performance than GPT2, thus using T5 model in most experiments. For OpenDialKG, since the ground-truth relations exist, we take them as an additional supervision signal as (Moon et al., 2019). Also, since we observe that there is only KG reasoning type in OpenDialKG, we do not use operation layer and checkpoints layer for the dataset. The hyperparameter settings are in Appendix C.

**6.4 Baselines** We compare our proposed DiffKG model with the state-of-the-art models on OpenDialKG reported in (Moon et al., 2019; Jung et al., 2020) and the state-of-the-art graph-based model on SMD (Yang et al., 2020; Gou et al., 2021) with their reported baselines including sequence-to-sequence models with and without attention (S2S and S2S+Attn) (Luong et al., 2015), pointer to unknown (Ptr-Unk) (Gulcehre et al., 2016),
### 6.5 Results

The results on SMD and OpenDialKG are shown in Table 2 and Table 3. On SMD dataset, we observe that DiffKG outperforms the baselines on BLEU by 11.4% (relative change of 16.04 and 14.4) and achieves comparable entity F1 scores with GLMP, GraphDialog and COMET-graph. DiffKG might not improve the entity F1 scores because that prior works group the text inside an entity together (e.g., “road block nearby” becomes a single word “road_block_nearby” in vocabularies). In contrast, we use a universal tokenizer so as to prevent heavy preprocessing and specialized vocabularies. This means that DiffKG can perform similarly with state-of-the-art to retrieve knowledge without a tokenizer specified for each dataset. On OpenDialKG dataset, we observe that DiffKG outperforms the baselines in terms of path@k scores and can simultaneously outperform T5 in terms of Entity F1 and BLEU. These demonstrate that DiffKG can retrieve accurate paths for reasoning and effectively incorporate reasoning into response generation.

We also investigate the results of SMD-Reasoning dataset as shown in Table 4. We find that DiffKG improves NoInfo by 16.6% and 44.4% F1 scores respectively on GPT2 and T5 models. This demonstrates that DiffKG can utilize knowledge effectively to improve the generation without access to information. In contrast, although FlatInfo gives similar performances as DiffKG on the SMD-Reasoning dataset, it cannot be run on OpenDialKG due to computational costs. More specifically, FlatInfo requires the knowledge graph to be transformed into sentences, which will result in at least a million tokens as the model inputs for OpenDialKG (since the number of triples is a million without designed subgraph sampling), which is not a practical number.
<table>
<thead>
<tr>
<th>Method</th>
<th>Schedule EM</th>
<th>Schedule F1</th>
<th>Navigation EM</th>
<th>Navigation F1</th>
<th>Weather EM</th>
<th>Weather F1</th>
<th>Inform EM</th>
<th>Inform F1</th>
<th>Selection EM</th>
<th>Selection F1</th>
<th>Extraction EM</th>
<th>Extraction F1</th>
<th>True/False EM</th>
<th>True/False F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT2-NoInfo</td>
<td>3.49</td>
<td>45.7</td>
<td>4.63</td>
<td>41.6</td>
<td>27.5</td>
<td>46.8</td>
<td>5.03</td>
<td>45.2</td>
<td>1.45</td>
<td>47.4</td>
<td>3.06</td>
<td>24.0</td>
<td>68.6</td>
<td>68.6</td>
</tr>
<tr>
<td>GPT2-DiffKG</td>
<td>9.30</td>
<td>53.0</td>
<td>9.65</td>
<td>47.6</td>
<td>34.4</td>
<td>56.5</td>
<td>8.04</td>
<td>50.8</td>
<td>0.00</td>
<td>48.5</td>
<td>31.6</td>
<td>53.5</td>
<td>56.9</td>
<td>56.9</td>
</tr>
<tr>
<td>T5-NoInfo</td>
<td>0.00</td>
<td>44.6</td>
<td>4.63</td>
<td>40.9</td>
<td>29.0</td>
<td>50.7</td>
<td>3.02</td>
<td>44.9</td>
<td>8.70</td>
<td>49.1</td>
<td>1.02</td>
<td>25.2</td>
<td>70.6</td>
<td>70.6</td>
</tr>
<tr>
<td>T5-DiffKG</td>
<td>20.9</td>
<td>63.8</td>
<td>19.3</td>
<td>61.9</td>
<td>48.1</td>
<td>68.1</td>
<td>18.1</td>
<td>61.7</td>
<td>11.6</td>
<td>62.4</td>
<td>50.0</td>
<td>73.5</td>
<td>70.6</td>
<td>70.6</td>
</tr>
</tbody>
</table>

Table 5: Detailed Evaluation Results of SMD-Reasoning dataset

<table>
<thead>
<tr>
<th>SMD-Reasoning</th>
<th>User: check the date and time of my doctor’s appointment (Reasoning Path: Doctor Appointment HasDate, HasTime, ToSelf → Tuesday, 11am, doctor appointment) DiffKG: inform 11 am tuesday doctor appointment</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMD-Reasoning</td>
<td>User: Car I need to get to a gas station, please show me the nearest one Assistant: There is Valero 7 miles away with moderate traffic on our way User: Alright, where is it located? (Reasoning Path: Gas Station IsTypeOf Valero HasAddress, ToSelf 200 Alester Ave, Valero) DiffKG: inform 200 Alester Ave Valero</td>
</tr>
<tr>
<td>OpenDialKG</td>
<td>Speaker A: Do you have any info on Toni Kroos? (Reasoning Path: Toni Kroos ~Player Statistics Player Statistics ~ Germany national football team) DiffKG: Toni Kroos is German footballer who plays for the Germany national football team.</td>
</tr>
</tbody>
</table>

6.6 Quantitative Analysis

To test the robustness of the methods towards accurately locating information, we shuffle the information order. This evaluation is to simulate the cases that extra information is arbitrarily added when deploying a dialogue system. Specifically, the order of the knowledge context for FlatInfo and the order of knowledge triples are changed during inference time. As shown in the last two rows in Table 4, the performance of FlatInfo drops while DiffKG remains about the same. This indicates that the slight superior performance of FlatInfo with the original order can come from the blackbox tricks to group the nearby knowledge in the inputs. When this implicit trick is broken down, the DiffKG shows much better robustness and performance.

To investigate the difficulty of each domain and reasoning type, we divide the results accordingly in Table 5. As presented in the domains part, the models achieve the highest EM and F1 on the weather domain. We conjecture the reason is that the weather domain includes more reasoning types (weather:4, navigate:3, schedule:2 as in Appendix A Table 9), thus reflecting more balanced reasoning ability. In the reasoning types part, we observe that true/false is less well coped by DiffKG; however, DiffKG improves the extraction. This shows that DiffKG can effectively check the existence of required knowledge and then query the database.

Regarding to the computational costs (on SMD-Reasoning dataset using T5 model), we found that DiffKG requires about 5.85GB memory during training and has 30ms inference latency. This could be an acceptable add-on memory usage and inference time compared to a model without knowledge reasoning (3.13GB; 50ms). Especially when a baseline like FlatInfo consumes much more (18.56GB; 50ms).

6.7 Qualitative Analysis

We visualize the generated examples and the symbolic reasoning path by DiffKG on SMD-Reasoning and OpenDialKG datasets in Table 6. The examples show that DiffKG can capture some naturally occurring phenomena in this dataset: (1) the KG reasoning path can be 1 to multiple hops; (2) the reasoning will diffuse to multiple paths (e.g., DiffKG simultaneously applies “HasDate”, “HasTime”, “ToSelf” to “Doctor Appointment”). Along with analyses in previous subsections, we observe that DiffKG can extract interpretable reasoning paths and generate corresponding outputs using reasonable computational costs.

However, even though DiffKG can maintain or improve performance while doing interpretable reasoning on any scaled KG, errors might happen in
some cases. As listed in Table 7, we found that across the datasets, the three main error types of DiffKG are: (1) unclear information requirement in the dataset, (2) incomplete reasoning ability but faithful response generation, and (3) correct reasoning but hallucinated response prediction. We argue that the first error type mainly comes from the mismatch among data points in the dataset and may not be able to be dealt with by models. The second error type indicates that the KG reasoning module sometimes cannot retrieve all the needed information. The third error type indicates that the module producing final output may not fully utilize the retrieved information. These three points might provide a direction for further improvement.

7 Conclusion and Future Work

For a dialogue system, an effective reasoning method over structured databases is important. In this work, we proposed DiffKG, an end-to-end model-agnostic method that does symbolic reasoning on any scale of KGs to enhance response generation. Experiments demonstrated that using DiffKG, models are able to generate responses with interpretable KG reasoning paths at a modest extra cost.

This work can be extended in various ways. While we solely consider efficient large-scale KG reasoning in dialogue generation, future work can incorporate domain fusion methods to consider the generalizability over domains or simultaneously use relation information. Moreover, since DiffKG is a simple large-scale structured knowledge-empowered transformer with flexible entity values, future work can extend it to dialogue generation that needs to do table and text mixed reasoning and that needs to do both KG reasoning and other goals such as personalized dialogues, storytelling, etc.

References


Julian Martin Eisenschlos, Maharshi Gor, Thomas Müller, and William W Cohen. 2021. Mate: Multi-


Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In EMNLP.


A Dataset Details

The statistics of datasets are in Table 8 and Table 9. The OpenDialKG dataset is under CC-BY-NC-4.0 license. These datasets can be used for research purposes.

<table>
<thead>
<tr>
<th>Data</th>
<th>Train</th>
<th>Validation</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMD</td>
<td>2425</td>
<td>302</td>
<td>304</td>
</tr>
<tr>
<td>OpenDialKG</td>
<td>10971</td>
<td>2351</td>
<td>2351</td>
</tr>
</tbody>
</table>

Table 8: The number of dialogues in each data split.

<table>
<thead>
<tr>
<th>Data</th>
<th>Schedule</th>
<th>Navigation</th>
<th>Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inform</td>
<td>364</td>
<td>1133</td>
<td>236</td>
</tr>
<tr>
<td>Selection</td>
<td>-</td>
<td>686</td>
<td>39</td>
</tr>
<tr>
<td>True/False</td>
<td>-</td>
<td>-</td>
<td>543</td>
</tr>
<tr>
<td>Extraction</td>
<td>173</td>
<td>474</td>
<td>214</td>
</tr>
</tbody>
</table>

Table 9: The statistics of SMD Reasoning dataset with respect to domains and reasoning types. Note that since reasoning types are classified on turn-level, the total number in this table is larger than in Table 8 that counted on dialogue-level.

B SMD KG Construction

We write a simple, automatic program to construct KGs for SMD dataset mapped from the original annotated tables.

For the schedule and navigation domains in SMD, we directly map their table attributes to the relations $R$ in our constructed KG. For the weather domain, we split each weather report into low temperature, high temperature, and weather. The resulting number of relations is 29, and the relations are listed in Table 10.

In the schedule and navigation domain, each item in the original database with multiple attributes are transformed to KG triples as (event/point-of-interest, attribute, attribute value), e.g., (tennis activity, HasTime, 7pm) in schedule domain or (Chevron, HasType, gas station) in navigation domain.

In the weather domain, we add additional entities named “ReportID$S$Digits$S$”, where $S$Digits$S$ will be replaced with an ID number. Each item in the original database is in the format: (item, location, $S$Location), (item, $S$Date, $S$Weather_report), where the $S$Weather_report contains multiple information not simultaneously needed. To make the KG of weather consistent with the KGs of schedule and navigation, we transform each item into (ReportID, location, $S$Location), (ReportID, HasDate, $S$date), (ReportID, HasWeather, $S$Weather), (ReportID, HasLowTemp, $S$LowTemp), (ReportID, HasHighTemp, $S$HighTemp), (ReportID, HasLowTemp, $S$LowTemp).

C Experiment Details

The hyperparameters we set for DiffKG are $d =$ the hidden size of the used pretrained transformer (T5-small: $d = 512$; GPT2-$d = 768$), $H = 5$, max norm=1.0, batch size=16, and gradient accumulation steps=2 for at most 50 epochs and train the model learning rate $\in \{ 5 \times 10^{-5}, 6.25 \times 10^{-5} \}$ (found that $6.25 \times 10^{-5}$ is better) without learning rate decay. Our experiments were single runs with random initialization and were not further fine-tuned.

D Computational Cost Analysis

As plotted in Figure 2, on SMD-Reasoning dataset, the consumed memory of FlatInfo is thrice the memory needed for DiffKG at training time, and its

<table>
<thead>
<tr>
<th>Domain</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule</td>
<td>HasTime, HasDate, HasParty, HasRoom, HasAgenda, IsTimeOf, IsDateOf, IsPartyOf, IsRoomOf, IsAgendaOf</td>
</tr>
<tr>
<td>Navigation</td>
<td>HasAddress, HasType, HasTraffic, HasDistance, IsAddressOf, IsTypeOf, IsTrafficOf, IsDistanceFrom</td>
</tr>
<tr>
<td>Weather</td>
<td>IsEqualTo, HasLocation, HasWeather, HasLowTemp, HasHighTemp, HasDate, IsWeatherOf, IsLowTempOf, IsHighTempOf, IsLocationOf, IsDateOf</td>
</tr>
</tbody>
</table>

Figure 2: The comparison of the consumed training memory and inference latency.
latency is about twice at inference time. The differ-
ence in inference latency is even larger with GPT2
as the backbone model. The reason is that the com-
putational cost of a causal language model such
as GPT2 largely depends on the input sequence
length, which is one of the main issues of FlatInfo.