An Interpretable Neuro-Symbolic Reasoning Framework for Task-Oriented Dialogue Generation

Shiquan Yang[1], **Rui Zhang**[2], **Sarah Erfani**[1], **and Jey Han Lau**[1]
[1]The University of Melbourne, [2]www.ruizhang.info
[1]{shiquan@student., sarah.erfani@, laujh@}unimelb.edu.au, [2]rayteam@yeah.net

Abstract

We study the interpretability issue of task-oriented dialogue systems in this paper. Previously, most neural-based task-oriented dialogue systems employ an *implicit* reasoning strategy that makes the model predictions uninterpretable to humans. To obtain a transparent reasoning process, we introduce neuro-symbolic to perform *explicit* reasoning that justifies model decisions by reasoning chains. Since deriving reasoning chains requires multi-hop reasoning for task-oriented dialogues, existing neuro-symbolic approaches would induce error propagation due to the one-phase design. To overcome this, we propose a two-phase approach that consists of a hypothesis generator and a reasoner. We first obtain multiple hypotheses, i.e., potential operations to perform the desired task, through the hypothesis generator. Each hypothesis is then verified by the reasoner, and the valid one is selected to conduct the final prediction. The whole system is trained by exploiting raw textual dialogues without using any reasoning chain annotations. Experimental studies on two public benchmark datasets demonstrate that the proposed approach not only achieves better results, but also introduces an interpretable decision process. Code and data: https://github.com/shiquanyang/NS-Dial.

1 Introduction

Neural task-oriented dialogue systems have enjoyed a rapid progress recently (Peng et al., 2020; Hosseini-Asl et al., 2020; Wu et al., 2020), achieving strong empirical results on various benchmark datasets such as SMD (Eric et al., 2017) and MultiWOZ (Budzianowski et al., 2018). However, most existing approaches suffer from the lack of explainability due to the black-box nature of neural networks (Doshi-Velez and Kim, 2017; Lipton, 2018; Bommasani et al., 2021), which may hurt the trustworthiness between the users and the system. For instance, in Figure 1, a user is asking for a hotel recommendation at a given location. The system performs reasoning on a knowledge base (KB) and incorporates the correct entity in the response. However, when the system fails to provide the correct entities, it would be difficult for humans to trace back the issues and debug the errors due to its intrinsic *implicit* reasoning nature. As a result, such system cannot be sufficiently trusted to be deployed in real-world products.

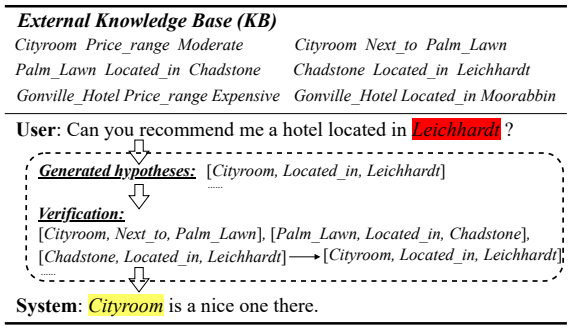

Figure 1: An example dialogue that incorporates external KB. The context entity (i.e., *Leichhardt*) and answer entity (i.e., *Cityroom*) are marked as Red and Yellow, respectively. The triple containing the context entity and answer entity is not directly stored in KB and should be derived by a reasoning chain formed by multiple KB triplets.

To achieve trustworthy dialogue reasoning, we aim to develop an interpretable KB reasoning as it's crucial for not only providing useful information (e.g., locations in Figure 1) to users, but also essential for communicating options and selecting target entities. Without interpretability, it's difficult for users to readily trust the reasoning process and the returned entities.

To tackle this challenge, we present a novel **N**euro-**S**ymbolic **Dial**ogue framework (**NS-Dial**) which combines representation capacities of neural networks and explicit reasoning nature of symbolic approaches (e.g., rule-based expert systems). Existing neuro-symbolic approaches (Vedantam et al.,

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4918 - 4935

2019; Chen et al., 2020) mostly employ a one-phase procedure where a tree-structured program composed of pre-defined human interpretable neural modules (e.g., attention and classification modules in Neural Module Networks (Andreas et al., 2016)) is generated to execute to obtain the final predictions. However, since the KB reasoning task involves a reasoning process spanning over multiple triplets in a diverse and large-scale KB, only generating and following a single program (i.e., a reasoning chain formed by KB triplets) is prone to error propagation where a mistake in one step could lead to a failure of the subsequent reasoning process and may result in sub-optimal performances.

To address this, we propose a two-phase procedure to alleviate the effects of error propagation by first generating and then verifying multiple hypotheses. Here, a hypothesis is in the form of a triplet containing an entity mentioned in dialogue context and an entity within KB, and their corresponding relation. The valid (i.e., correct) hypothesis is the one that contains the entity mentioned in the ground-truth response. Once we obtain multiple hypothesis candidates during the generation phase, we employ a reasoning engine for verifying those hypotheses. For instance in Figure 1, given the user query "*Can you recommend me a hotel located in Leichhardt?*", in order to find the valid hypothesis, the hypothesis generator obtains multiple candidates e.g., *[Cityroom, Located_in, Leichhardt]* and *[Gonville_Hotel, Located_in, Leichhardt]*. The reasoning engine will then construct proof trees to verify them, e.g., for the first hypothesis *[Cityroom, Located_in, Leichhardt]*, it can be verified with the following reasoning chain in the KB: *[Cityroom, Next_to, Palm_Lawn]* $\rightarrow$ *[Palm_Lawn, Located_in, Chadstone]* $\rightarrow$ *[Chadstone, Located_in, Leichhardt]*. The whole framework is trained end-to-end using raw dialogues and thus does not require additional intermediate labels for either the hypothesis generation or verification modules.

To summarize, our contributions are as follows:

- We introduce a novel neuro-symbolic framework for interpretable KB reasoning in task-oriented dialogue systems.
- We propose a two-phase "generating-and-verifying" approach which generates multiple hypotheses and verifies them via reasoning chains to mitigate the error-propagation issue.
- We conduct extensive experimental studies on two benchmark datasets to verify the effectiveness of our proposed model. By analyzing the generated hypotheses and the verifications, we demonstrate our model's interpretability.

2 Related Work

Task-Oriented Dialogue Traditionally, task-oriented dialogue systems are built via pipeline-based approaches where task-specific modules are designed separately and connected to generate system responses (Chen et al., 2016; Zhong et al., 2018; Wu et al., 2019a; Chen et al., 2019a; Huang et al., 2020). In another spectrum, many works have started to shift towards end-to-end approaches to reduce human efforts (Bordes et al., 2017; Lei et al., 2018; Madotto et al., 2018; Moon et al., 2019; Jung et al., 2020). Lei et al. (2018) propose a two-stage sequence-to-sequence model to incorporate dialogue state tracking and response generation jointly in a single sequence-to-sequence architecture. Zhang et al. (2020) propose a domain-aware multi-decoder network (DAMD) to combine belief state tracking, action prediction and response generation in a single neural architecture. Most recently, the success of large-scale pre-trained language models (e.g., BERT, GPT-2) (Devlin et al., 2018; Radford et al., 2019) has spurred a lot of recent dialogue studies starting to explore large-scale pre-trained language model for dialogues (Wolf et al., 2019; Zhang et al., 2019). In task-oriented dialogue, Budzianowski and Vulić (2019) use GPT-2 to fine-tune on MultiWOZ dataset for dialogue response generation. Peng et al. (2020) and Hosseini-Asl et al. (2020) employed a single unified GPT-2 model jointly trained for belief state prediction, system action and response generation in a multi-task fashion. However, most existing approaches cannot explain why the model makes a specific decision in a human understandable way. We aim to address this limitation and introduce interpretability for dialogue reasoning in this study.

Neuro-Symbolic Reasoning Neuro-Symbolic reasoning has attracted a lot of research attentions recently due to its advantage of exploiting the representational power of neural networks and the compositionality of symbolic reasoning for more robust and interpretable models (Andreas et al., 2016; Hu et al., 2017; Hudson and Manning, 2018; Vedantam et al., 2019; Chen et al., 2019b; Vedantam et al., 2019; van Krieken et al., 2022). The main difference between neuro-symbolic vs. pure

neural networks lies in how the former combines basic rules or modules to model complex functions. Rocktäschel and Riedel (2017) propose a neuro-symbolic model that can jointly learn subsymbolic representations and interpretable rules from data via standard back-propagation. In visual QA, Andreas et al. (2016) propose neural module networks to compose a chain of differentiable modules wherein each module implements an operator from a latent program. Yi et al. (2018) propose to discover symbolic program trace from the input question and then execute the program on the structured representation of the image for visual question answering. However, these approaches cannot be easily adapted to task-oriented dialogues due to the error propagation issue caused by multi-hop reasoning on large-scale KBs. Thus, we aim to bridge this gap by developing a neuro-symbolic approach for improving task-oriented dialogues.

3 Preliminary

In this work, we focus on the problem of task-oriented dialogue response generation with KBs. Formally, given the dialogue history X and knowledge base B, our goal is to generate the system responses Y word-by-word. The probability of the generated responses can be written as:

$$p(Y|X,B) = \prod_{t=1}^{n} p(y_t|X,B,y_1,y_2,...,y_{t-1}) \quad (1)$$

where y_t is the t-th token in the response Y. The overall architecture is shown in Figure 2. We start by introducing the standard modules in our system and then explain the two novel modules afterward.

3.1 Dialogue Encoding

We employ pre-trained language model BERT (Devlin et al., 2019) as the backbone to obtain the distributed representations for each token in the dialogue history. Specifically, we add a $[CLS]$ token at the start of the dialogue history to represent the overall semantics of the dialogue. The hidden states $H_{enc} = (h_{CLS}, h_1, ..., h_M)$ for all the input tokens $X = ([CLS], x_1, ..., x_M)$ are computed using:

$$H_{enc} = \text{BERT}_{\text{enc}}(\phi^{emb}(X)) \quad (2)$$

where M is the number of tokens in the dialogue history, ϕ^{emb} is the embedding layer of BERT.

3.2 Response Generation

To generate the system response, we first utilize a linear layer to project H_{enc} to $H'_{enc} = (h'_{CLS}, h'_1, ..., h'_M)$ that are in the same space of the decoder. We initialize the decoder with h'_{CLS}. During decoding timestep *t*, the model utilizes the hidden state $h_{dec,t}$ to attend H'_{enc} to obtain an attentive representation $h'_{dec,t}$ via standard attention mechanism. We then concatenate $h_{dec,t}$ and $h'_{dec,t}$ to form a context vector C and project it into the vocabulary space $\mathcal{V}$:

$$C = [h_{dec,t}, h'_{dec,t}] \quad (3)$$

$$P_{vocab,t} = \text{Softmax}(U_1 C) \quad (4)$$

where U_1 is a learnable linear layer, $P_{vocab,t}$ is the vocabulary distribution for generating the token y_t.

Next, we aim to estimate the KB distribution $P_{kb,t}$, i.e., the probability distribution of entities in the KB, in an interpretable way and fuse $P_{vocab,t}$ and $P_{kb,t}$ for generating the final output tokens. We follow See et al. (2017) and employ a soft-switch mechanism to fuse $P_{vocab,t}$ and $P_{kb,t}$ to generate output token y_t. Specifically, the generation probability $p_{gen} \in [0,1]$ is computed from the attentive representation $h'_{dec,t}$ and the hidden state $h_{dec,t}$:

$$p_{gen} = \sigma(U_2([h'_{dec,t}, h_{dec,t}])) \quad (5)$$

where σ is sigmoid function, U_2 is a linear layer. The output token y_t is generated by greedy sampling from the probability distribution $P(w)$:

$$P(w) = p_{gen} P_{vocab,t} + (1 - p_{gen}) P_{kb,t} \quad (6)$$

We next describe how to obtain the KB distribution $P_{kb,t}$ in details using the two novel modules we proposed, i.e., hypothesis generator and hierarchical reasoning engine.

4 Neuro-Symbolic Reasoning For Task-Oriented Dialogue

To compute the KB distribution $P_{kb,t}$, we present two novel modules: hypothesis generator (HG) and hierarchical reasoning engine (HRE). We take the context vector C (Equation 3) as the input of HG module and generate K hypotheses $\mathbb{H}$, each of which are then fed into the HRE module to generate the logical reasoning chains and their belief scores. The estimated belief scores are then served as $P_{kb,t}$, giving us a distribution over the entities in the KB. Next, we describe how each component works in detail and explain how they interact with each other for generating $P_{kb,t}$.

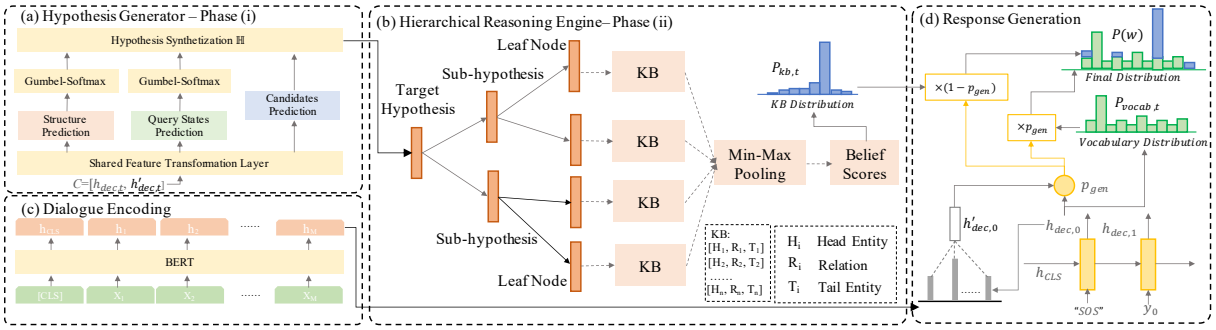

Figure 2: Illustration of the overall architecture: (a) hypothesis generator generating a set of synthesized hypotheses; (b) reasoning engine used to verify the generated hypotheses; (c) dialogue encoding; (d) response generation.

4.1 Hypothesis Generator

Let a hypothesis be a 3-tuple of the form "$[H, R, T]$", where H and T are the head and tail entities, and R is the relation between entities. In this paper, we are interested in three types of hypotheses including the H-Hypothesis, T-Hypothesis, and R-Hypothesis. The H-Hypothesis is the structure where the tail entity T and relation R are inferred from the context and the head entity H is unknown (which needs to be answered using the KB), and it takes the form "$[\triangleright, R, T]$". In a similar vein, the T-Hypothesis and R-Hypothesis have unknown tail entity T and relation R, respectively. The goal of the Hypothesis Generator module is to generate hypotheses in this triple format which will later be verified by the Hierarchical Reasoning Engine.

Intuitively, a hypothesis can be determined by its content and structure. The structure indicates the template form of the hypothesis while the content fills up the template. For instance, the H-Hypothesis has its template form of "$[\triangleright, R, T]$" and the content that needs to be realised includes candidate entities (i.e., "$\triangleright$"), and query states (i.e., the tail "T" and relation entities "R"). To this end, we employ a divide-and-conquer strategy to jointly learn three sub-components: structure prediction, query states prediction, and candidates prediction. Next, we describe each sub-component in details.

Structure Prediction (SP) The goal of the structure prediction module is to determine the structure of the hypothesis (i.e., H/T/R-Hypothesis) based on the context. For example in Figure 1, one might expect an H-Hypothesis at timestep 0. Specifically, SP uses a shared-private architecture to predict the hypothesis type. It first takes the context vector C (Equation 3) as input and utilizes a shared transformation layer between all the three sub-components to learn task-agnostic feature h_{share}:

$$h_{share} = W_2(\text{LeakyReLU}(W_1 C)) \tag{7}$$

where W_1 and W_2 are learnable parameters (shared by the structure prediction, query states prediction and candidate prediction components) and LeakyReLU is the activation function.

The shared layer can be parameterised with complicated neural architectures. However, to keep our model simple, we use linear layers which we found to perform well in our experiments. SP next uses a private layer on top of the shared layer to learn task-specific features for structure prediction:

$$h^{sp}_{private} = W_4(\text{LeakyReLU}(W_3 h_{share})) \tag{8}$$

where W_3 and W_4 are learnable parameters. For ease of presentation, we define the private feature transformation function as:

$$\mathcal{F}^{\star} : h_{share} \rightarrow h^{\star}_{private} \tag{9}$$

where $\star$ denotes any of the three sub-components. To obtain the predicted hypothesis structure, a straightforward approach is to apply softmax on $h^{sp}_{private}$. However, this will break the differentiability of the overall architecture since we perform sampling on the outcome and pass it to the neural networks. To avoid this, we utilize the Gumbel-Softmax trick (Jang et al., 2017) over $h^{sp}_{private}$ to get the sampled structure type:

$$\text{I}_{sp} = \text{Gumbel-Softmax}(h^{sp}_{private}) \in \mathbb{R}^3 \tag{10}$$

where I_{sp} is a one-hot vector and the index of one element can be viewed as the predicted structure. In this paper, we define 0 as H-Hypothesis, 1 as T-Hypothesis and 2 as R-Hypothesis.

Query States Prediction (QSP) Query states are the tokens in hypothesis that need to be inferred from the dialogue history. For example, one might want to infer relation R=*Located_in* and tail

T=*Leichhardt* based on the history in Figure 1. Therefore, the goal of the query states prediction is to estimate the state information (e.g., T and R in H-Hypothesis) of hypothesis. Specifically, QSP takes the shared feature h_{share} as the input and next applies the private feature transformation function followed by Gumbel-Softmax to obtain the state tokens of hypothesis using:

$$h^{qsp,k}_{private} = \mathcal{F}^{qsp,k}(h_{share}) \quad (11)$$

$$\mathrm{I}^{k}_{qsp} = \text{Gumbel-Softmax}(h^{qsp,k}_{private}) \in \mathbb{R}^{n} \quad (12)$$

where n is the number of tokens (entities and relations) in the KB, $k \in \{0,1\}$, I^{0}_{qsp} and I^{1}_{qsp} are two one-hot vectors where their corresponding tokens in KB serve as the state tokens of the hypothesis.

Candidates Prediction (CP) To generate the final hypotheses, we need multiple candidates to instantiate the structure of the hypothesis except the state tokens, e.g., *Cityroom* or *Gonville_Hotel* as candidate head entities H in Figure 1. To this end, we utilize an embedding layer ϕ^{emb}_{cp} to convert all the tokens in the KB to vector representations. We then compute a probability distribution over all the KB tokens using:

$$\mathrm{P}_i = \text{Sigmoid}(\phi^{emb}_{cp}(K_i) \odot h_{share}) \quad (13)$$

where K_i is the i-th token in KB, ϕ^{emb}_{cp} is the embedding layer of CP, P_i is the probability of the i-th token to be candidate, $\odot$ denotes inner-product. We use sigmoid instead of softmax as we find that softmax distribution is too "sharp" making the probability between different tokens are hard to differentiate for sampling multiple reasonable candidates.

Hypothesis Synthesizing The final hypotheses $\mathbb{H}$ are composed by combining the outputs of the three sub-components as follows: (i) We generate the hypothesis template according to the predicted structure type. For example, if SP predicts a structure type 0 which denotes H-Hypothesis, the model will form a template of "$[\triangleright, R, T]$"; (ii) We next instantiate the state tokens in the hypothesis sequentially by using the outputs of QSP module. For example, if the output tokens of QSP are "*Located_in*" (k=0) and "*Leichhardt*" (k=1), the hypothesis will become $[\triangleright, \textit{Located\_in}, \textit{Leichhardt}]$; (iii) Finally, we instantiate the candidate (i.e., $\triangleright$) with the top-K (K =5 in our best-performing version) entities selected from P. If the top-2 highest probability tokens are *Cityroom* and *Gonville_Hotel*, the model will instantiate two hypotheses *[Cityroom, Located_in, Leichhardt]*, *[Gonville_Hotel, Located_in, Leichhardt]*.

4.2 Hierarchical Reasoning Engine

With the hypotheses generated by HG module, we next aim to verify them via logical reasoning chains. Inspired by Neural Theorem Provers (Rocktäschel and Riedel, 2017), we develop chain-like logical reasoning with following format:

$$\alpha,\ (H, R, T) \leftarrow (H, R_n, Z_n) \wedge \cdots \wedge (Z_1, R_1, T) \quad (14)$$

where α is a weight indicating the *belief* of the model on the target hypothesis $[H, R, T]$, and the right part of the arrow is the reasoning chain used to prove that hypothesis, and R_i and Z_i are relations and entities from the KB. The goal is to find the proof chain and the confidence α for a given hypothesis. To this end, we introduce a neural-network based hierarchical reasoning engine (HRE) that learns to conduct chain-like logical reasoning. At a high level, HRE recursively generates multiple levels of sub-hypotheses using neural networks that form a tree structure as shown in Figure 2. Next, we describe how this module works in details.

The module takes the output hypotheses from the HG module as input. Each hypothesis serves as one target hypothesis. To generate the reasoning chain in Equation 14, the module first finds sub-hypotheses of the same format as the target in the hypothesis space. The sub-hypotheses can be viewed as the intermediate reasoning results to prove the target. One straightforward approach is to use neural networks to predict all the tokens in the sub-hypotheses (2 heads, 2 tails and 2 relations). However, this can lead to extremely large search space of triples and is inefficient. Intuitively, sub-hypotheses inherit from the target hypothesis and sub-hypotheses themselves are connected by bridge entities. For example, *[Uber,office_in,USA]* can be verified by two sub-hypotheses *[Uber,office_in,Seattle]* and *[Seattle,a_city_of,USA]*, *Uber* and *USA* are inherited from the target and *Seattle* is the bridge entity between sub-hypotheses. Motivated by this, we propose to reduce the triple search complexity by constraining the sub-hypotheses. Specifically, given target $[H, R, T]$, we generate sub-hypotheses of the format $[H, R_1, Z]$,$[Z, R_2, T]$, where Z is the bridge entity, R_1 and R_2 are relations to be predicted. Therefore, the goal of the neural networks has been reduced to predict three tokens (2 relations and 1 bridge entity). Formally, HRE predicts the vector representation of bridge entity as follows:

$$h_H, h_R, h_T = \phi_{cp}^{emb}(H), \phi_{cp}^{emb}(R), \phi_{cp}^{emb}(T) \quad (15)$$

$$h_Z = W_6(\text{LeakyReLU}(W_5[h_H, h_R, h_T])) \quad (16)$$

where $[h_H, h_R, h_T]$ are the concatenation of the representations of tokens in target hypothesis, h_Z is the vector representation of bridge entity Z. The prediction of h_{R_1} and h_{R_2} uses the same architecture in Equation 16 and the difference is that they use different linear layers for the feature transformation. Note that h_Z denotes a KB token in the embedding space. We can decode the token by finding the nearest KB token to h_Z in vector space. More details on the token decoding can be found in Appendix A. Upon obtaining h_Z, h_{R_1}, h_{R_2}, the module generates the two sub-hypotheses in vector representations. Next, the module iteratively takes each of the generated sub-hypothesis as input and extend the proof process by generating next-level sub-hypotheses in a depth-first manner until the maximum depth D has been reached.

Belief Score To model confidence in different reasoning chains, we further measure the semantic similarities between each triple of the leaf node and triples in the KB, and compute the belief score α_m of the m-th hypothesis $\mathbb{H}_m$:

$$\alpha_m = \min_{\forall i \in U} \max_{\forall j \in V} e^{-d_j(Leaf_i, KB_j)} \quad (17)$$

where $Leaf_i$ is the representation (concatenation of H, R, T) of the i-th leaf node in the proof tree (DFS manner), KB_j is the representation of the j-th triple in KB, U=[0,...,u-1], V=[0,...,v-1] where u and v are the number of leaf nodes and KB triples correspondingly, d is the distance metric. In general, any distance function can be applied and we adopt Euclidean distance in our implementation since we found that it worked well in our experiments. All the triples in the leaf nodes form the reasoning chain for the input hypothesis as in Equation 14. The hypotheses $\mathbb{H}$ coupled with the belief α form our KB distribution $P_{kb,t}$. More details can be found in Appendix B. Intuitively, the belief score can be viewed as the likelihood of the hypothesis contains the correct entity. If the hypothesis is valid (i.e., contains the correct answer entity), it should have a high likelihood and thus encourage to generate more proper reasoning chains based on the triples stored in the KB.

Training We apply two loss functions to train the whole architecture end-to-end. The first loss function $\mathcal{L}_{gen}$ is for the final output. We use a cross-entropy loss over the ground-truth token and the generated token from the final distribution $P(w)$. The second loss $\mathcal{L}_{cp}$ is for the candidates prediction (CP) module in the hypotheses generator. We apply binary cross-entropy loss over the output distribution for each KB token (Equation 13) and their corresponding labels. The labels for each KB token are computed as follows:

$$Label_i = \begin{cases} 1, & K_i = y_t \\ 0, & K_i \neq y_t \end{cases} \quad (18)$$

where K_i is the i-th token in the KB and y_t is the ground-truth output at timestep t. The final loss $\mathcal{L}$ is calculated by:

$$\mathcal{L} = \gamma_g * \mathcal{L}_{gen} + \gamma_c * \mathcal{L}_{cp} \quad (19)$$

where γ_g and γ_c are hyper-parameters and we set them to 1 in our experiments.

Dataset	Domains	Train	Dev	Test
SMD	Navigate,Weather,Schedule	2425	302	304
MultiWOZ 2.1	Restaurant,Attraction,Hotel	1839	117	141

Table 1: Statistics of SMD and MultiWOZ 2.1.

5 Experiments

5.1 Datasets

To evaluate the effectiveness and demonstrate the interpretability of our proposed approach, we conduct experiments on two public benchmark datasets for task-oriented dialogue in this paper, SMD (Eric et al., 2017) and MultiWOZ 2.1 (Budzianowski et al., 2018). We use the partitions created by Eric et al. (2017); Madotto et al. (2018) and Qin et al. (2020) for SMD and MultiWOZ, respectively. Statistics of the datasets are presented in Table 1. In the Appendix E, we present several additional results on a large-scale synthetic dataset to demonstrate our model's multi-hop reasoning capability under complex KB reasoning scenarios.

5.2 Baselines

We compare our model with the following state-of-the-art baselines on KB reasoning in task-oriented dialogues: (1) Mem2Seq (Madotto et al., 2018): employs memory networks to store the KB and combine pointer mechanism to either generate tokens from vocabulary or copy from memory; (2) GLMP (Wu et al., 2019b): uses a global-to-local pointer mechanism to query the KB during decoding; (3) DF-Net (Qin et al., 2020): employs

	SMD					MultiWOZ 2.1				
Model	BLEU	F1	Navigate F1	Weather F1	Calendar F1	BLEU	F1	Restaurant F1	Attraction F1	Hotel F1
Mem2Seq	12.6	33.4	20.0	32.8	49.3	6.6	21.6	22.4	22.0	21.0
GLMP	13.9	60.7	54.6	56.5	72.5	6.9	32.4	38.4	24.4	28.1
GraphDialog	14.2	61.1	56.4	56.9	72.1	6.7	34.1	39.2	27.8	29.6
DF-Net	14.4	62.7	57.9	57.6	73.1	9.4	35.1	40.9	28.1	30.6
Ours (D=1)	14.9	63.8	60.1	58.7	75.0	9.7	36.5	42.0	29.7	32.8
Ours (D=3)	**15.6***	**64.5***	**60.3***	**59.2***	**75.6***	**10.6***	**37.2***	**42.6***	**30.6***	**33.7***
Ours (D=5)	14.5	63.5	59.4	57.9	74.8	9.3	36.2	41.7	28.8	31.5

Table 2: Main results. D denotes the maximum depth of HRE module. We run each experiment 5 times with different random seeds and report the average results. * denotes that the improvement of our framework over all baselines are statistically significant with $p < 0.05$ under t-test. Following Qin et al. (2020), we report *Navigate, Weather, Calendar* on SMD and *Restaurant, Attraction, Hotel* on MultiWOZ for per-domain results.

shared-private architecture to capture both domain-specific and domain-general knowledge to improve the model transferability; (4) GraphDialog (Yang et al., 2020): incorporates graph structural information obtained from sentence dependency parsing results for improving KB reasoning accuracy and response generation quality. Detailed experimental settings are included in Appendix C.

5.3 Main Results

Following prior work (Eric et al., 2017; Madotto et al., 2018; Wu et al., 2019b), we adopt the *BLEU* and *Entity F1* metrics to evaluate the performance of our framework. The results on the two datasets are shown in Table 2. As we can see, our framework consistently outperforms all the previous state-of-the-art baselines on all datasets across both metrics. Specifically, on MultiWOZ dataset, our model achieves more than 2% absolute improvement in Entity F1 and 1.2% improvement in BLEU over baselines. The improvement in Entity F1 indicates that our model enhances KB reasoning, while the increase in BLEU suggests that the quality of the generated responses has been improved. The same trend has also been observed on SMD dataset. This indicates the effectiveness of our proposed framework for task-oriented dialogue generation.

5.4 Model Interpretability

To demonstrate our framworks's interpretability, we investigate the inner workings of our framework. As shown in Figure 3, given the dialogue history "*Can you recommend me a restaurant near Palm_Beach?*", the generated response is "*There is a Golden_House.*". During the 3rd timestep, our model has successfully predicted an appropriate *H-Hypothesis* with *Located_in* and *Palm_Beach* as its state tokens. Our model further instantiates five concrete hypotheses and computes their belief scores leveraging the reasoning engine, respectively. As we can see from the table, our model successfully generates five reasonable hypotheses and scores them correctly (with highest score for the oracle KB entity *Golden_House*). The proof process for the highest score hypothesis is shown in Figure 3. The verification procedure generated by the HRE module has a depth of 3 and the reasoning chaining used to verify the target hypothesis is: *[Golden_House, Next_to, Preston_Market]* $\rightarrow$ *[Preston_Market, Located_in, Williamstown]* $\rightarrow$ *[Williamstown, Located_in, Herb_Garden]* $\rightarrow$ *[Herb_Garden, Located_in, Palm_Beach]*. This indicates that our framework has successfully utilized the KB information to support the reasoning process *explicitly* to reach a correct conclusion. More examples and error analyses can be found in the Appendix (Appendix E.4 and F).

5.5 Ablation Study

We ablate each component in our framework to study their effectiveness on both datasets. The results are shown in Table 3. Specifically, 1) w/o HRE denotes that we simply use the probability in candidates prediction (CP) module (Equation 13) as the KB distribution without using the scores from the reasoning engine. 2) w/o BERT denotes that we use standard GRU as encoder instead of BERT. 3) w/o Soft-switch denotes that we simply sum the KB distribution and vocabulary distribution without using a soft gate. As we can see from the table, all the individual components have notably contributed to the overall performance of

Dialogue history: *Can you recommend me a restaurant near Palm_Beach?* **Predicted response**: *There is a Golden_House.*

Structure Type	State Tokens	Top-5 Candidate Tokens	Set of Generated Hypotheses	Belief Scores
H-Hypothesis	"*Located_in* (k=0)"	"*The_Hotpot*"	"*[The_Hotpot, Located_in, Palm_Beach]*"	0.13
		"*Hookey_Park*"	"*[Hookey_Park, Located_in, Palm_Beach]*"	0.07
		"*Golden_House*"	"*[Golden_House, Located_in, Palm_Beach]*"	1.00
	"*Palm_Beach* (k=1)"	"*Rose_Lands*"	"*[Rose_Lands, Located_in, Palm_Beach]*"	0.08
		"*Princes_Gardens*"	"*[Princes_Gardens, Located_in, Palm_Beach]*"	0.05

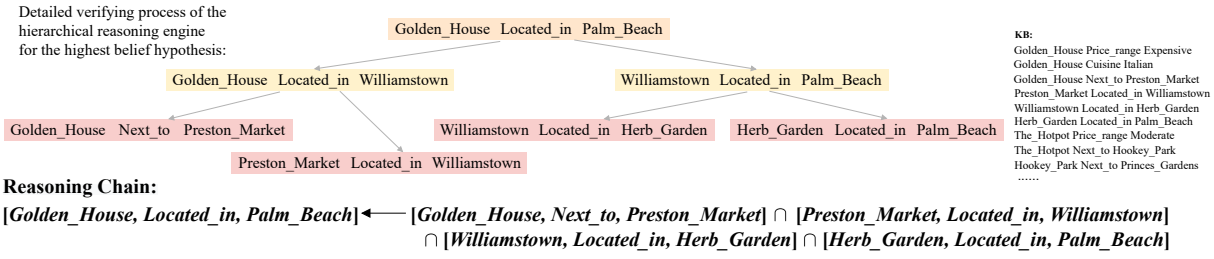

Figure 3: Example of inner workings of the hypothesis generator and hierarchical reasoning engine for generating *Golden_House* in the response given dialogue history *Can you recommend me a restaurant near Palm_Beach?*. Our model has performed a 4-hop reasoning to verify the target hypothesis *[Golden_House, Located_in, Palm_Beach]*.

	SMD		MultiWOZ 2.1	
Model	F1 (%)	Δ	F1 (%)	Δ
Ours (Full model)	**64.5**	-	**37.2**	-
- w/o HRE	59.4	5.1	30.5	6.7
- w/o BERT	61.3	3.2	33.4	3.8
- w/o Soft-switch	62.0	2.5	35.1	2.1

Table 3: Ablation studies on two benchmark datasets.

	SMD		MultiWOZ 2.1	
Model	Original F1 (%)	Unseen F1 (%)	Original F1 (%)	Unseen F1 (%)
GLMP	60.7	55.3	32.4	23.9
GraphDialog	61.1	55.7	34.1	25.4
DF-Net	62.7	57.2	35.1	26.5
Ours (Full)	**64.5**	**61.1**	**37.2**	**32.8**

Table 4: Generalization test results on two datasets.

our framework. Specifically, when removing HRE module, the performance has decreased substantially (more than 5% absolute drop), which confirms that the effectiveness of the proposed hierarchical reasoner module.

5.6 Generalization Capability

We further investigate the generalization ability of our model under unseen settings. In the original dataset released by prior works, the entity overlap ratio between the train and test split is 78% and 15.3% for MultiWOZ 2.1 and SMD, respectively. To simulate unseen scenario, we construct a new dataset split that reduces the entity overlap ratio to 30% for MultiWOZ 2.1 and 2% for SMD between the train and test split, which is a more challenging setting for all the models. More details of the construction process can be found in Appendix D. We re-run all the baselines with their released codes and our model on the new data split and report the results in Table 4. As we can see, the performance drops significantly for all systems on both datasets. However, our model degrades less compared to other systems, showing that it has better generalisation capability under unseen scenarios. This also verifies that neuro-symbolic approach has the advantage of better generalisation ability which has also been confirmed by many other studies (Andreas et al., 2016; Rocktäschel and Riedel, 2017; Minervini et al., 2020).

5.7 Human Evaluation

Following prior work (Qin et al., 2020), we also conduct human evaluations for our framework and baselines from three aspects: *Correctness*, *Fluency*, and *Humanlikeness*. Details about the scoring criterions can be found in Appendix H. We randomly select 300 different dialogue samples from the test set and ask human annotators to judge the quality of the responses and score them according to the three metrics ranging from 1 to 5. We train the annotators by showing them examples to help them

Model	Correct	Fluent	Humanlike
GLMP	4.01	3.78	3.25
GraphDialog	4.15	4.19	3.40
DF-Net	4.16	4.25	3.54
Ours (Full model)	**4.41**	**4.28**	**3.59**
Human	4.83	4.65	4.57
Agreement	75%	69%	71%

Table 5: Human evaluation results.

understand the criteria and employ Fleiss' kappa (Fleiss, 1971) to measure the agreement across different annotators. The results are shown in Table 5. As we can see, our model outperforms all baselines across all the three metrics, consistent with our previous observations using automatic evaluations.

6 Conclusion

In this paper, we propose an explicit and interpretable Neuro-Symbolic KB reasoning framework for task-oriented dialogue generation. The hypothesis generator employs a divide-and-conquer strategy to learn to generate hypotheses, and the reasoner employs a recursive strategy to learn to generate verification for the hypotheses. We evaluate our proposed framework on two public benchmark datasets including SMD and MultiWOZ 2.1. Extensive experimental results demonstrate the effectiveness of our proposed framework, as well being more interpretable.

7 Ethical Considerations

For the human evaluation in this paper, we recruit several annotators on Amazon Mechanical Turk from English-speaking countries. We pay the annotators USD$0.15 for each annotation task. Each task can be finished on average in 1 minute, which amounts to $9.0 per hour that is above the US federal minimum wage ($7.25). To ensure the quality of the human evaluation results, we perform quality control in a few ways. First, the annotators will be shown our scoring standards (Appendix H) before their tasks, and are asked to follow them. If the task is not done properly, either as determined by expert judgements (we recruit 3 native English speakers to validate the results of the Turkers' annotations) or there are obvious patterns such as constantly giving the same score for all tasks, we remove their annotations. We also compute agreement score to check for the consistency among the annotators.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 39–48.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Kohd, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the opportunities and risks of foundation models.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In ICLR.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2–how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. arXiv preprint arXiv:1907.05774.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 5016–5026.

Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019a. Semantically

conditioned dialog response generation via hierarchical disentangled self-attention. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3696–3709.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V Le. 2019b. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In International Conference on Learning Representations.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In International Conference on Learning Representations.

Yun-Nung Chen, Dilek Hakkani-Tür, Gökhan Tür, Jianfeng Gao, and Li Deng. 2016. End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding. In Interspeech, pages 3245–3249.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, pages 37–49.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. Psychological bulletin, 76(5):378.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. arXiv preprint arXiv:2005.00796.

Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, pages 804–813.

Xinting Huang, Jianzhong Qi, Yu Sun, and Rui Zhang. 2020. Semi-supervised dialogue policy learning via stochastic reward estimation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 660–670, Online. Association for Computational Linguistics.

Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. arXiv preprint arXiv:1803.03067.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In ICLR.

Jaehun Jung, Bokyung Son, and Sungwon Lyu. 2020. AttnIO: Knowledge Graph Exploration with In-and-Out Attention Flow for Knowledge-Grounded Dialogue. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 3484–3497, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics.

Zachary C Lipton. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue, 16(3):31–57.

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Mem2Seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1468–1478.

Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. 2020. Learning reasoning strategies in end-to-end differentiable proving. In International Conference on Machine Learning, pages 6938–6949. PMLR.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Building task bots at scale with transfer learning and machine teaching. arXiv preprint arXiv:2005.05298.

Libo Qin, Xiao Xu, Wanxiang Che, Yue Zhang, and Ting Liu. 2020. Dynamic fusion network for multi-domain end-to-end task-oriented dialog. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6344–6354, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9.

Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end differentiable proving. arXiv preprint arXiv:1705.11040.

Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end differentiable proving. In Advances in Neural Information Processing Systems, volume 30.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1073–1083.

Emile van Krieken, Erman Acar, and Frank van Harmelen. 2022. Analyzing differentiable fuzzy logic operators. Artificial Intelligence, 302:103602.

Ramakrishna Vedantam, Karan Desai, Stefan Lee, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. 2019. Probabilistic neural symbolic models for interpretable visual question answering. In International Conference on Machine Learning, pages 6428–6437. PMLR.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. arXiv preprint arXiv:1901.08149.

Chien-Sheng Wu, Steven Hoi, Richard Socher, and Caiming Xiong. 2020. Tod-bert: pre-trained natural language understanding for task-oriented dialogue. arXiv preprint arXiv:2004.06871.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019a. Transferable multi-domain state generator for task-oriented dialogue systems. arXiv preprint arXiv:1905.08743.

Chien-Sheng Wu, Richard Socher, and Caiming Xiong. 2019b. Global-to-local memory pointer networks for task-oriented dialogue. In ICLR.

Shiquan Yang, Rui Zhang, and Sarah Erfani. 2020. GraphDialog: Integrating graph knowledge into end-to-end task-oriented dialogue systems. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1878–1888.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B Tenenbaum. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. arXiv preprint arXiv:1810.02338.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 9604–9611.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. arXiv preprint arXiv:1911.00536.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive dialogue state tracker. arXiv preprint arXiv:1805.09655.

A Details on Token Decoding in HRE

Given the vector representations of the generated sub-hypotheses in hierarchical reasoning engine module, we utilize the similarity-based approach to decode the symbolic representations of those sub-hypotheses. Specifically, given a generated sub-hypotheses $[h_H, h_R, h_T]$, where h_H, h_R and h_T are the vector representations for the head entity, relation and tail entity correspondingly. To decode the symbolic representations for the head, relation and tail entities, we use:

$$\arg\min_{\forall i} \quad \|\phi(K_i) - h_H\|^2. \tag{20}$$

$$\arg\min_{\forall j} \quad \|\phi(K_j) - h_R\|^2. \tag{21}$$

$$\arg\min_{\forall k} \quad \|\phi(K_k) - h_T\|^2. \tag{22}$$

where *i*, *j* and *k* are the indices for the head entity, relation and tail entity in the vocabulary, K_i, K_j, K_k denotes the i-th, j-th, k-th token of the KB, $\phi(K_i)$ denotes the embedding of the i-th token. Through this, we can decode the generated sub-hypotheses and obtain their explicit symbolic representations.

B Details on KB Distribution Calculation

We extract the KB distribution $P_{kb,t}$ at timestep t from the generated hypotheses and their corresponding belief scores as follows. For instance, if the generated hypothesis $[H, R, T]$ is an H-Hypothesis with a belief score α, we extract the candidate token of the H-Hypothesis which is H and then pair H with the belief score α, where α is viewed as the probability of the token H to be selected as the output at timestep t. We conduct this for all the generated hypotheses and their corresponding belief scores from the HG and HRE modules. Finally, all the candidate tokens paired with their belief scores form the $P_{kb,t}$ at timestep t.

C Experimental Settings

The dimensionality of the embedding and the decoder RNN hidden units are 128 and embeddings are randomly initialized. The dropout ratio is selected from [0.1, 0.5]. We use Adam (Kingma and Ba, 2014) optimizer to optimize the parameters in our model and the learning rate is selected from [$1e^{-3}$,$1e^{-4}$]. For the encoder, we fine-tune the BERT-base-uncased model from HuggingFace's library with an the embedding size of 768 with 12 layers and 12 heads. The maximum depth D of the HRE module is selected from [1,5], the maximum number of candidates K in CP module is selected from [1,10], and the temperature of Gumbel-Softmax is 0.1. All hyper-parameters are selected according to the validation set, and we repeat all the experiments 5 times with different random seeds and report the average results.

D Details on Unseen Setting

We construct new dataset splits both on SMD and MultiWOZ 2.1 to simulate unseen scenarios for testing the generalization ability of all the models. Specifically, we construct the new dataset split as follows: We first extract all the KB entities that appeared in the dialogue responses and accumulate the percentage of samples for each KB entity. Second, we rank all the entities according to their percentage of samples in a decreasing order. Next, we split the KB entity set into train entities and test entities by accumulating the total percentages of samples. Finally, we iterate each sample in the dataset and assign it to train or test split by checking whether the entity in the response belong to the train entities or test entities. In this way, we obtain a new dataset split for both SMD and MultiWOZ 2.1, which has an entity overlap ratio of 2% and 30%, respectively, between train and test split (overlap ratio in the original SMD and MultiWOZ 2.1 are 15.3% and 78%, respectively).

The dataset statistics for the unseen splits are shown in Table 6 and Table 7:

Dataset	Train	Dev	Test
SMD	1850	311	870
MultiWOZ 2.1	1472	252	373

Table 6: Statistics of Unseen Dataset for SMD and MultiWOZ 2.1.

Dataset	Ent. Overlap Standard	Ent. Overlap Unseen	$\Delta\downarrow$
SMD	15.3%	2%	13.3%
MultiWOZ 2.1	78%	30%	48%

Table 7: Entity Overlap Ratio Comparisons Between Unseen Split and Original Split for SMD and MultiWOZ 2.1. Entity Overlap Ratio = |Train Entities $\bigcap$ Test Entities| / |Total Entities|.

E Additional Experiments

We find that KB reasoning for most existing task-oriented dialogue datasets are quite simple, for the most part only requiring that only one or two hop reasoning over the KB in order to answer the user's request successfully. To further test our model and baseline models' multi-hop reasoning capability under complex reasoning scenarios, we develop a large-scale multi-domain synthetic dataset consisting dialogues requiring multi-hop reasoning over KBs. This is similar in spirit to bAbI dataset, and we hope that this dataset will continue to be used with other dialogue benchmarks in future studies. We will release this dataset upon publication. Next, we describe how we construct the dataset in details and show the experimental results performed on it.

E.1 Dataset Construction

As is shown in Figure 4, each sample in the dataset consists of several rounds of dialogues. We generate the questions and answers of the dialogues by randomly sample template utterances with placeholders (e.g., @*movie*, @*director*, @*location*) indicating the types of KB entities to be instantiated to form the complete utterances. To simulate a natural conversation between user and system under different scenarios (i.e., restaurant booking, hotel reservation, movie booking), we designed 18 different types of question-answer templates. For example, *movie to director* denotes that the user requests the director given the movie name, *location to theatre* denotes the user requires theatre information given the location. For each conversation, we randomly select several different types of question-answer templates sequentially to form the skeleton of the whole dialogue. To ensure the coherent of the dialogue flow, we provide the guided next types for each question-answer template. For instance, if the current sampled question-answer type is *location to restaurant*, the guided next types will be randomly sampled from *restaurant to price*, *restaurant to cuisine* etc. Thus, we can ensure the generated dialogue turns more coherent in terms of semantics to simulate a real conversation as much as possible.

For each conversation, we generate 3 or 4 rounds of dialogues following the existing work such as *SMD* and *MultiWOZ 2.1*. At each round of the dialogue, we randomly select a question-answer template and instantiate the placeholders in the template with the corresponding types of KB entities. If there are multiple entities in the KB satisfy the types indicated by the placeholders, we randomly sample one to implement the template. In this way, we can increase the diversity of the generated data. For instance, if the question template is *Is there any restaurant located in @district?*, the possible sets of entities in the KB for the placeholder @*district* might include multiple location entities in the KB such as *vermont*, *blackburn* etc. We randomly sample one of them to replace the placeholder and generate a final sentence. If we sample *vermont*, the implemented sentence will be *Is there any restaurant located in the vermont?*.

To make the generated dialogue utterances more natural as human conversations, we further randomly replace the KB entities in the sentence with pronouns such as *it*, *they* etc, provided that the entities have been mentioned in previous dialogue turns. Thus, it requires the model to overcome the co-reference resolution to arrive at the correct answer which increases the difficulty. For example, *Who is the director of the movie mission impossible?* will be rephrased as *Who is the director of it?* if the movie name *mission impossible* has been mentioned in the dialogue history.

For movie domain, we employ the KB used in the well-known *WikiMovie* dataset. For hotel and restaurant domain, we use the KB provided in the *MultiWOZ 2.1* dataset. For each employed KB, we further extend it by adding information such as hierarchies of locations to enrich the KB in order to make it suitable for testing multi-hop reasoning capability. For example, if the KB contains a hotel entity *love lodge*, we add different levels of location information to support multi-hop KB reasoning. For instance, we add location information such as *love_lodge next_to lincoln_park*, *lincoln_park is_within waverley_district*, *waverley_district located_in grattan_county*. Thus, if the user asked about the hotel located in *grattan_county*, it requires the model to conduct multi-hop reasoning over the KB to know that *love_lodge*is located in *grattan_county*. Through this, we make our synthetic dataset suitable for multi-hop reasoning tasks over KB under task-oriented dialogue scenarios. The location information we utilized in the synthetic dataset are obtained from the Wikipedia and the official website of famous cities around the world.

External Knowledge Base (KB)	
Cityroom Price_range Moderate	*Cityroom Stars 3*
Cityroom Next_to Palm_Lawn	*Palm_Lawn Located_in Chadstone*
Chadstone Located_in Leichhardt	*Gonville_Hotel Stars 4*
Gonville_Hotel Price_range Expensive	*Gonville_Hotel Located_in Moorabbin*

User: Can you recommend me a hotel located in *Leichhardt* ?
System: *Cityroom* is a nice one there.
User: How much does it cost there ?
System: It has a *moderate* price range.
User: What is the rating of it ?
System: It is a *3 star* hotel.

Figure 4: An example dialogue from the hotel domain of the synthetic dataset. The first turn of the dialogue requires a 3-hop reasoning over the KB to get the correct entity Cityroom given the location information Leihhardt. The second and third turn of the dialogue require single-hop reasoning over KB to get the correct entity.

E.2 Dataset Statistics

The detailed statistics of the synthetic dataset are shown in Table 8 and Table 9:

Domain	Train	Dev	Test
Movie	7219	1645	1667
Hotel	7115	1631	1639
Restaurant	7131	1672	1684
Total	21465	4948	4990

Table 8: Statistics of synthetic dataset. Numbers in the table are the number of instances for each category.

E.3 Experimental Results

Evaluation Metrics. We use the same metrics as on *SMD* and *MultiWOZ 2.1* dataset includes *BLEU* and *Entity F1* for performance evaluation.

Results. The results on the three domains are shown in Table 10, 11, 12. For each domain, we evaluate the model performance on different subsets of the test data, i.e., 1-hop, 2-hop and >=3-hop. Specifically, we group the test data into three different subsets according to the KB reasoning length for obtaining the ground-truth entity. For instance, 2-hop denotes that the KB entity mentioned in the response needs 2-hop reasoning over the KB. As we can see from the tables, our proposed model consistently outperforms all the baselines by a large margin across all the domains and KB reasoning lengths. We also observe that all the models' performance decrease monotonously as the KB reasoning path length increases, suggesting that KB reasoning with longer range is challenging for all the tested models. However, our framework has less performance degradation compared to all the baselines, and the performance gap between our framework and the baselines has become larger when the length of KB reasoning increases, which demonstrates that our framework has better generalization ability especially under longer KB reasoning paths compared to those baselines.

E.4 Example Outputs

We show the generated hypotheses and the proof trees in our framework as shown in Table 13 and Figure 5. As we can see, our model can successfully obtain the correct entities from the KB. Moreover, our framework can formulate sensible hypotheses and generate reasonable proof procedures which can help us gain some insights about the inner workings of our model.

F Error Analysis

We conduct error analysis on both *SMD* and *MultiWOZ 2.1* to provide insights in our framework for future improvements. We randomly sample 100 dialogues from each test set and analysis both the generated responses and the inner procedures. The errors have four major categories: 1) structure errors, 2) query states errors, 3) candidates errors, 4) belief score errors. For example, given dialogue history "*Where is a nearby parking_garage?*", the generated response is "*5671_barringer_street is 1_mile away.*" and the ground-truth is "*The nearest one is palo_alto_garage, it's just 1_mile away.*". The generated hypotheses by HG module at timestep 0 are "[*5671_barringer_street*, *distance*, *1_mile*], [*whole_foods*, *distance*, *1_mile*], [*home*, *distance*, *1_mile*], [*valero*, *distance*, *1_mile*], [*pizza_chicago*, *distance*, *1_mile*]". The predicted structure type is a H-Hypothesis with a relation "*distance*" and tail entity "*1_mile*". However, a reasonable hypothesis should be an H-Hypothesis such as "[*palo_alto_garage*, *poi_type*, *parking_garage*]" with "*poi_type*" (place of interest) and "*parking_garage*" as relation and tail. This illustrates the challenge of the task, and we plan to further improve our framework include SP, QSP, CP and HRE in future work.

F.1 Structure Prediction Errors

Given the dialogue history "*Find me a nearby shopping mall.*", the generated response is "*mid-town_shopping_center is 3_miles away.*" and

	Movie			Hotel			Restaurant		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
Hop=1	27238	5985	5998	14321	3472	3482	14386	2107	2117
Hop=2	6401	1472	1507	3351	594	614	4527	564	609
Hop>=3	5359	1508	1530	3328	514	524	4545	562	593
Total	38998	8965	9035	21000	4580	4620	23458	3233	3319

Table 9: Detailed statistics of the synthetic dataset with respect to the number of hops needed by KB reasoning. Numbers in the table are the number of dialogue turns for each category. Hop=k denotes that the KB reasoning path length for the entity in the dialogue response is k.

	Movie Domain							
	1-Hop		**2-Hop**		**Hop>=3**		**All**	
	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1
Mem2Seq	25.6	68.9	21.8	60.8	19.5	49.2	23.9	62.0
GLMP	30.1	77.2	28.7	72.9	27.1	61.5	28.3	73.2
GraphDialog	29.2	76.6	25.6	69.1	24.7	60.6	27.2	71.6
DF-Net	30.6	77.4	29.5	71.6	28.9	62.1	30.3	73.5
Ours (Full model)	**33.2**	**82.6**	**31.3**	**80.4**	**30.7**	**74.9**	**32.7**	**80.6**

Table 10: Experimental results on the movie domain of the synthetic dataset.

	Hotel Domain							
	1-Hop		**2-Hop**		**Hop>=3**		**All**	
	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1
Mem2Seq	14.4	79.8	13.1	71.2	11.4	68.6	13.2	75.4
GLMP	21.3	85.5	19.8	79.4	18.9	76.2	21.0	82.9
GraphDialog	20.6	83.8	19.1	78.8	18.8	75.9	19.3	81.0
DF-Net	22.1	86.7	19.9	80.2	19.5	76.8	21.5	83.2
Ours (Full model)	**23.3**	**92.4**	**21.3**	**89.6**	**20.7**	**87.8**	**22.1**	**91.6**

Table 11: Experimental results on the hotel domain of the synthetic dataset.

	Restaurant Domain							
	1-Hop		**2-Hop**		**Hop>=3**		**All**	
	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1
Mem2Seq	19.0	79.8	17.3	69.4	12.4	66.3	17.0	73.7
GLMP	22.0	90.4	19.1	83.7	18.4	80.4	20.9	86.1
GraphDialog	23.2	89.9	21.2	82.1	20.6	79.8	21.4	85.0
DF-Net	24.5	91.5	23.0	84.2	21.1	81.0	23.3	87.3
Ours (Full model)	**26.8**	**96.7**	**24.4**	**93.1**	**22.7**	**92.2**	**25.1**	**94.2**

Table 12: Experimental results on the restaurant domain of the synthetic dataset.

Structure Type	State Tokens	Top-5 Candidate Tokens	Generated Hypotheses	Belief Scores
H-Hypothesis	"*located_in* (k=0)"	"*Shipping_News*"	"*[Shipping_News, located_in, Springfield]*"	0.15
		"*Vaudeville*"	"*[Vaudeville, located_in, Springfield]*"	0.17
		"*Brown_Eyes*"	"*[Brown_Eyes, located_in, Springfield]*"	0.13
	"*Springfield* (k=1)"	"*Oakland*"	"*[Oakland, located_in, Springfield]*"	1.00
		"*Coburg*"	"*[Coburg, located_in, Springfield]*"	0.04

Table 13: Example outputs on the movie domain of synthetic dataset. Dialogue history: "*I'm looking for a theatre in the Springfield district.*". Generated response: "*Sure I have found a Oakland for you.*". Detailed model working process when generating Oakland in the response is shown above.

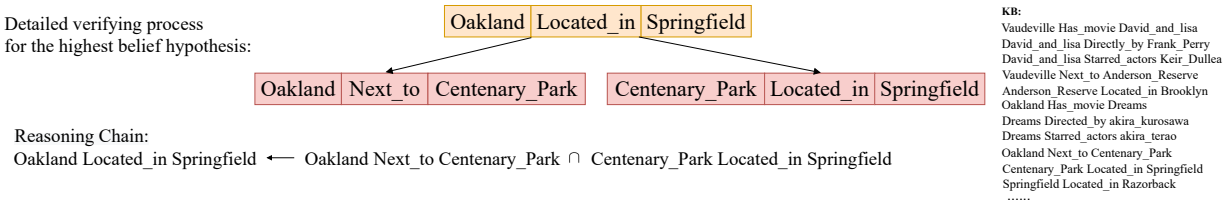

Figure 5: Proof tree generated by HRE module for the highest score hypothesis *[Oakland, Located_in, Springfield]* in Table 13. The red parts are the predicted bridge entities and the blue parts are the predicted relations for the sub-hypotheses via neural networks. In this case, the model performs 2-hop reasoning (the two leaf node triples) to find the correct KB entity for generating the response.

the ground-truth response is "*The closest shopping_center is midtown_shopping_center 3_miles away.*". The generated hypotheses by HG module at decoding timestep 0 are "*[home, address, 5671_barringer_street]*, *[home, address, 481_amaranta_ave]*, *[home, address, midtown_shopping_center]*, *[home, address, 5_miles]*, *[home, address, 1_miles]*". As we can see, the predicted structure type is an T-Hypothesis with an head entity "*home*" and relation "*address*". However, a reasonable structure type should be an H-Hypothesis such as "*[midtown_shopping_center, poi_type, shopping_center]*" which the model has mispredicted to a T-Hypothesis.

F.2 Query States Prediction Errors

Given the dialogue history "*What time is dinner tonight?*", the generated response is "*Dinner is at 11pm.*" and the ground-truth response is "*Your dinner is 5pm tonight.*". The generated hypotheses in HG module for decoding timestep 3 are "*[dinner, traffic_info, 11pm]*, *[dinner, traffic_info, 7pm]*, *[dinner, traffic_info, 3pm]*, *[dinner, traffic_info, Monday]*, *[dinner, traffic_info, Tuesday]*". As we can see, the SP and QSP modules have successfully predicted a sensible T-Hypothesis with "*dinner*" as head entity. However, the predicted relation "*traffic_info*" is not reasonable according to context where an expected relation would be relations such as "time".

F.3 Candidates Prediction Errors

Given the dialogue history "*What time is my doctor_appointment?*", the generated response is "*Your doctor_appointment is on Thursday.*" and the ground-truth response is "*Your doctor_appointment is scheduled for Friday at 11am.*". The generated hypotheses by HG for decoding timestep 4 are "*[doctor_appointment, date, Thursday]*, *[doctor_appointment, date, Monday]*, *[doctor_appointment, date, 7pm]*, *[doctor_appointment, date, Tuesday]*, *[doctor_appointment, date, 2pm]*". As we can see, our framework has correctly predicted a T-Hypothesis and state tokens. However, the predicted top-5 candidate tokens don't include the ground-truth date which is *Friday*, which might affect the final response generation.

F.4 Belief Score Prediction Errors

Given the dialogue history "*Where is home?*", the generated response is "*Home is at 819_alma_st.*" and the ground-truth is "*Home is where is your heart, at 56_cadwell_street.*" The generated hypotheses for decoding timestep 3 are "*[home, address, 819_alma_st]*, *[home, address, 56_cadwell_street]*, *[home, address, 6_miles]*, *[home, address, 611_ames_ave]*, *[home, address, 3_miles]*". As we can see, our framework has predicted a sensible T-Hypothesis with "*home*" as head entity and "*address*" as relation. Also, the CP module has predicted top-5 candidate tail

entities which include the ground-truth *56_cadwell_street*. But the HRE module ranked "*[home, address, 819_alma_st]*" highest with a score of 0.78 while the ground-truth one "*[home, address, 56_cadwell_street]*" is only ranked the second highest with a score of 0.41, which indicates that there is still room for improvements for the HRE module. We are interested in continually improving our framework include all the modules in future work.

G Discussions

G.1 Why not use search-based techniques for generating reasoning chains?

This is an alternative approach to our learning-based method. However, search-based approach cannot be jointly learnt end-to-end with other modules in our framework, and thus may face error propagation and credit assignment issues like in the traditional pipeline-based task-oriented dialogue approaches. In this work, we want to explore the possibility of learning end-to-end the logical reasoning chain directly from the dialogues. Also, the time complexity of search-based approach is approximately $O(n^k)$, where n is the average degree of nodes in the external knowledge base, k is the number of reasoning hops. In other words, the time complexity tends to have polynomial growth (when $k > 1$); and it's worse when the reasoning complexity (k) increases (exponential). In contrast, when the number of KB nodes increases it only impacts the size of the input embedding layer in our framework (Equation 15), and the efficiency can be further improved by leveraging modern accelerating hardware such as GPU (which search-based approaches cannot).

G.2 Why sample with Gumbel-Softmax instead of directly applying argmax in Hypothesis Generator and Hierarchical Reasoning Engine modules?

Argmax function is non-differentiable which hinders our aim of end-to-end differentiability of the whole system. We tried utilizing REINFORCE (reward is obtained by comparing predicted entities with ground-truth entities) to mitigate this issue. However, we find that the results of using argmax+REINFORCE is worse than using Gumbel-Softmax. By checking the sampled tokens from Gumbel-Softmax, we find that it can generate reasonable tokens (Figure 3 in the main paper, state tokens etc.), since we have set the temperature parameter of Gumbel-Softmax to 0.1 which is a close approximation to argmax.

G.3 Why not expand the KB using KB completion methods and then use semantic parsing to query KB?

In this work, we are interested in developing an end-to-end trainable framework with explainable KB reasoning. Semantic parsing is one possible alternative. However, when adapting to our own dataset, it requires further annotations for fine-tuning which is costly and time-consuming, and might be not feasible for large-scale datasets. Also, it might induce the error propagation issue since the different modules (KB completion, semantic parsing, dialogue encoding and response generation etc.) are not jointly learnt.

G.4 KB scale.

The average nodes of KB for each sample in the training data is 63.5 for *SMD* and 57.6 for *MultiWOZ*. The average number of relations is 5.5 for *SMD* and 9.4 for *MultiWOZ*.

H Human Evaluation Details

The *Fluency* of the predicated responses is evaluated according to the following standards:

- 5: The predicted responses contain no grammar errors or repetitions at all.
- 4: Only one grammar error or repetition appeared in the generated responses.
- 3: One grammar error one repetition, or two grammar errors, or two repetitions are observed in the responses.
- 2: One grammar error two repetitions, or one repetition two grammar errors, or three grammar errors, or three repetitions appeared in the generated responses.
- 1: More than three inappropriate language usages with regard to grammar errors or repetitions are observed in the responses.

The *Correctness* is measured as follows:

- 5: Provide the correct entities.
- 4: Minor mistakes in the provided entities.
- 3: Noticeable errors in the provided entities but acceptable.

- 2: Poor in the provided entities.
- 1: Wrong in the provided entities.

The *Humanlikeness* is measured as:

- 5: 100% sure that the sentences are generated by a human, not by system.
- 4: 80% chance that the sentences are generated by a human.
- 3: Cannot tell whether the sentences is generated by a human or system, 50% for human and 50% for system.
- 2: 20% chance that the sentences are generated by a human.
- 1: Totally impossible that the sentences are generated by a human.