AttnIO: Knowledge Graph Exploration with In-and-Out Attention Flow for Knowledge-Grounded Dialogue

Jaehun Jung[1,2] **Bokyung Son**[1,3*] **Sungwon Lyu**[1*]

[1]Kakao Enterprise Corporation
[2]Department of Computer Science, Seoul National University
[3]Department of Linguistics, Seoul National University
sharkmir1@snu.ac.kr, {meta.mon, james.ryu}@kakaoenterprise.com

Abstract

Retrieving the proper knowledge relevant to conversational context is an important challenge in dialogue systems, to engage users with more informative response. Several recent works propose to formulate this knowledge selection problem as a path traversal over an external knowledge graph (KG), but show only a limited utilization of KG structure, leaving rooms of improvement in performance. To this effect, we present *AttnIO*, a new dialog-conditioned path traversal model that makes a full use of rich structural information in KG based on two directions of attention flows. Through the attention flows, *AttnIO* is not only capable of exploring a broad range of multi-hop knowledge paths, but also learns to flexibly adjust the varying range of plausible nodes and edges to attend depending on the dialog context. Empirical evaluations present a marked performance improvement of *AttnIO* compared to all baselines in OpenDialKG dataset. Also, we find that our model can be trained to generate an adequate knowledge path even when the paths are not available and only the destination nodes are given as label, making it more applicable to real-world dialogue systems.

1 Introduction

One of the milestone challenges in conversational AI is to engage users with a more informative and knowledgeable response, rather than merely outputting generic sentences. For instance, given a user's utterance saying "I'm a big fan of Steven Spielberg", it would be more engaging to respond "My favorite movie is his science fiction film A.I.", rather than "I like him too.". An external source of knowledge such as knowledge graph (KG) can play a crucial role here, as it could help the conversational agent with informative paths, such as "Steven Spielberg, directed, A.I., has genre, Science Fiction".

The above mentioned motivation gave rise to a conspicuous need for path retrieval model on KG, which can learn to traverse a path consisting of proper entities and relations to mention in the next response, given the dialog context. Previous approaches to this knowledge selection problem rely on either an RL-based agent (Liu et al., 2019) or a recurrent decoder (Moon et al., 2019), which greedily selects the most proper entity to traverse regarding its previous decision. Despite their novelty, we find several rooms of improvement from the previous works, to move toward a more fine-grained modeling of knowledge path retrieval for dialogue systems.

First, a model could make use of the rich relational information residing at the neighborhood of each node on KG. Typically, the number of entities in KG is large, while the numbers of each entity's usage in actual dialogues are small. Thus leveraging the neighborhood information of each entity in knowledge graph could be crucial, to overcome the sparsity of entity usage and learn proper representation of entities and relations.

Also, we find that the range of knowledge paths plausible for a response to a given dialog may vary, depending on the dialog context and user intent. In response to a closed question such as "Who directed the movie A.I.?", there could be only one or two knowledge paths valid as answer. On the contrary, in response to an open question such as "Do you know Steven Spielberg?", there could be a variety of knowledge paths natural enough to carry on the conversation. Therefore, a model should be able to choose the range of entities to attend, depending on the characteristics of a given dialog.

Lastly, one should note that it is practically hard

[*]Equal contribution.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 3484–3497,
November 16–20, 2020.

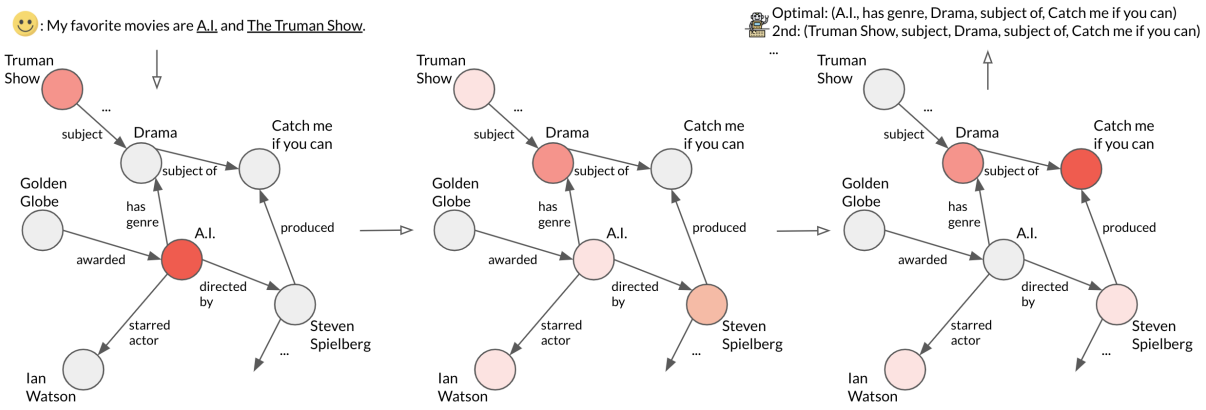

Figure 1: **Path decoding process** of AttnIO. By propagating attention values at each step rather than selecting one node to traverse, AttnIO can choose between focusing on small number of entities or attending to multiple relevant neighbors, depending on the dialog context.

to gather a large-scale dialog-KG path parallel corpus, fully annotated with all entities and relations comprising each path. Retrieving the initial, and final entities of the KG path is relatively easier, as it only requires post-processing the {query, response} pairs in dialog. Therefore, it would be more desirable if a model could be trained to traverse a proper knowledge path, only with the destination nodes provided as label.

To this end, we propose *AttnIO* (**Attention In**-flow and **O**ut-flow), a novel KG path traversal model that overcomes all challenges stated above. Aside to the conventional textual encoder which encodes dialog history and user utterance, *AttnIO* models the KG traversal mechanism into two subprocesses: *incoming attention flow*, and *outgoing attention flow*. Inspired by Attention Flow (Xu et al., 2018), the two attention flows explore KG by propagating the attention value at each node to its reachable neighbor nodes, as shown in Figure 1. The attention propagation mechanism enables our model to start exploring KG from multiple entities (*A.I.*, and *The Truman Show*), then find out an intermediate node *Drama* relevant to both movies, and end the multi-hop reasoning by arriving at *Catch me if you can*. Such a complex interaction between entities cannot be modeled by a greedy decoder, limited to consider only an optimal node at each decoding step. In addition, our model provides better interpretation of its path reasoning process, by visualizing the attention distribution of nodes and edges at each step. Lastly, we consider our model in a more challenging, but more realistic setting of path retrieval task, where no ground-truth path is available for supervision, but only the final destination nodes are given. Even in this setting, we find that *AttnIO* can be trained to infer a proper knowledge path for the input dialog.

In summary, our contributions are as follows: (1) We suggest a novel path traversal model *AttnIO*, achieving state-of-the-art performance in dialog-conditioned knowledge path retrieval task on the OpenDialKG dataset. (2) We demonstrate that *AttnIO* can be trained even in a challenging setting where only the destination nodes are given, and show through both qualitative and quantitative analysis that the quality of paths generated from this setting does not fall behind that of the all-path supervision setting. (3) Through visualizing the attention distribution at each decoding step, we show that our model possesses better interpretability over the path reasoning process.

2 Related Works

Recently, lots of research effort have been devoted to grounding dialogue systems on structured-knowledge embedded in knowledge graphs. These works can be broadly classified into two categories, depending on the range of exploration over candidate knowledge. The first line of works, namely *breadth-centric approaches*, tend to focus on augmenting dialog context with entity representations, by aggregating their *shallow* (i.e., 1-hop or 2-hop) neighborhood information from an external knowledge graph (Young et al., 2018; Liu et al., 2018; Parthasarathi and Pineau, 2018; Chen et al., 2019). Zhou et al. (2018) suggest to encode an auxiliary knowledge vector by attentively reading all 1-hop relations of each initial entity that appears in user's utterance. Zhang et al. (2019) extends the previous

work's knowledge encoding scheme to 2-hop relations, encoding all initial entities and their 1-hop neighbors with two independent attention mechanisms. While these works are successful in contextualizing each entity with various relations in KG, they lack in retrieving small set of focused knowledge paths relevant to the dialog, or generalizing to multi-hop relations. We extend these approaches by suggesting a new framework that can be generalized to an arbitrary length of traversal, and dynamically updating entity features at each decoding step to facilitate multi-hop relational inference.

On the other hand, the second line of works resort to *depth-centric search* over candidate knowledge paths. Rather than augmenting entity representation with shallow but wide range of knowledge, they concentrate on traversing only a specific range of entities and relations directly usable for response generation. Liu et al. (2019) formulate the knowledge selection problem as Partially Observed Markov Decision Process, employing a policy network to traverse KG. Meanwhile, Moon et al. (2019) suggest a recurrent path decoder that relies on a hidden state vector to choose the next entity among reachable nodes. Although these models are competent at inferring multi-hop relations, their discrete selection mechanism neglects rich relational information of nodes and edges they did not explicitly choose to traverse. To complement the weakness, AttnIO does not select an optimal node in advance; rather, it first propagates attention to all reachable entities, and then decode an optimal path from the output attention distribution.

Our work is also closely motivated from recent techniques suggested in the domain of knowledge graph completion tasks. To compensate for weak representation power of translative embedding (Bordes et al., 2013; Trouillon et al., 2016) and convolution-based embedding (Dettmers et al., 2018; Nguyen et al., 2018), several models have adopted graph neural networks (GNN), encoding structural information into entity embedding (Shang et al., 2019; Nathani et al., 2019). Other works perform traversal-based inference in node-prediction tasks based on reinforcement learning (Das et al., 2018; Lin et al., 2018), or attention propagation (Xu et al., 2018). We extend these previous works by adopting graph neural network and attention propagation for the dialog-conditioned path generation problem.

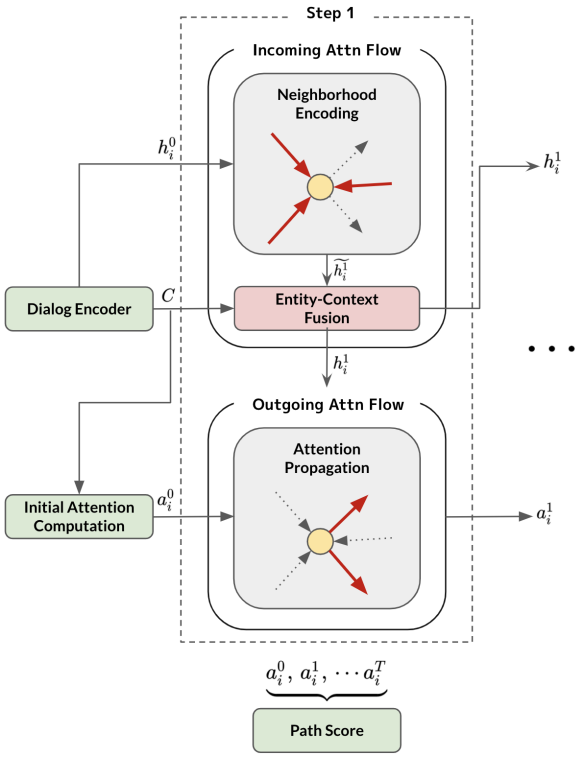

Figure 2: **AttnIO Model Overview**

3 Proposed method

3.1 Overview

We denote the external knowledge graph as $G_{KG} = V_{KG} \times R_{KG}$, with nodes as entities and edges as relations between a pair of entities. We denote $G_{v,n} \subseteq G_{KG}$ as a subgraph containing all nodes and edges reachable in less then or equal to n-hops, starting from vertex v. Also, we define $\overrightarrow{N}_i$ as a set of incoming neighbor nodes of v_i, i.e. nodes possessing edges toward v_i, and $\overleftarrow{N}_i$ as a set of outgoing neighbor nodes of v_i.

Figure 2 illustrates the overview of AttnIO's path generation process. Given the input multi-turn dialog sequence $\{s_1, \cdots, s_n\}$ and the set of entities $V_{init} = \{e_1, \cdots, e_m\}$ appearing in the user's last utterance s_n, AttnIO starts from encoding the input dialog into a fixed-size context vector. It also constructs the dialog-relevant subgraph $G_{input} = \bigcup_i G_{e_i,T}$, where T is a hyperparameter indicating the maximal length of path to traverse.

At each decoding step $t = 1, \cdots, T$, the *incoming attention flow* iteratively updates the KG entity features by attentively aggregating rich relational features from their incoming neighbor nodes. Then, the *outgoing attention flow* propagates the attention value of each node to its outgoing neighbor nodes, yielding the node attention

distribution a_i^t and edge attention distribution a_{ij}^t as step t's output. We show that each candidate entity path $P_v = \{P_v^{(0)}, \cdots, P_v^{(T)}\}$ and the relation path $P_r = \{P_r^{(1)}, \cdots, P_r^{(T)}\}$ can easily be ranked from these output attention distributions, in Section 3.4.1.

3.2 Dialog Encoder

AttnIO encodes the input multi-turn dialog into a fixed-size contextual representation. Specifically, we employ state-of-the-art textual representation from ALBERT (Lan et al., 2019), to effectively capture the context and intent of the user's utterance. We concatenate maximum of 3 last utterances in the dialog, and put it as input to the pretrained ALBERT. We use the final layer's hidden representation of [CLS] token, as it is typically considered to be an approximation of the sequence context. We denote this context vector as $\mathbf{C}$, in the following section.

Note that our architecture does not require a specific type of textual encoder, and ALBERT can be replaced with any sequence encoder such as bidirectional RNN. For a fair comparison with previous work, we conduct an ablation study on ALBERT by replacing it with bidirectional GRU (Section 4.1).

3.3 Incoming Attention Flow

In order to find better entity representation, the Incoming Attention Flow iteratively updates each entity feature $\mathbf{h}_j$ for all $v_j \in G_{input}$, by aggregating v_j's neighbor information. Recently suggested message-passing mechanism of *graph attention networks* (GAT) from Veličković et al. (2018) is suitable for this, as it learns to encode each node by selectively attending over its neighbors. Since GAT does not take account of edge features and hence may lose useful relational information integral in KG, we extend the attention-based message passing scheme of GAT into relational graphs. At each decoding step t, the Incoming Attention Flow computes message from entity v_i to v_j as follows:

$$\mathbf{m}_{ij} = \mathbf{W}_m \left[\mathbf{h}_i + \mathbf{r}_{ij}\right] \tag{1}$$

where $\mathbf{h}_i$ denotes the feature of v_i at step t, and $\mathbf{r}_{ij}$ denotes the relation feature assigned to the edge between v_i and v_j. Then, the new node feature $\widetilde{\mathbf{h}}'_j$ for the next time step $t+1$ is computed as an attention-based weighted sum of messages from all incoming neighbor nodes of v_j:

$$\widetilde{\mathbf{h}}'_j = \sum_{i \in \overrightarrow{N_j}} a_{ij} \mathbf{m}_{ij} \tag{2}$$

The attention a_{ij} is computed by applying softmax over v_j's all incoming neighbor nodes:

$$\begin{aligned} a_{ij} &= \underset{i \in \overrightarrow{N_j}}{\texttt{softmax}}(\alpha_{ij}), \\ \alpha_{ij} &= \sigma\left((\mathbf{W}_Q \mathbf{h}_j)^T (\mathbf{W}_k(\mathbf{h}_i + \mathbf{r}_{ij}))\right) \end{aligned} \tag{3}$$

where σ denotes LeakyReLU non-linearity.

In addition, we extend our attentive aggregation scheme to multi-headed attention, which helps to jointly attend to information from different representation subspaces of incoming messages (Vaswani et al., 2017). Thus our message aggregation mechanism in Eq.2 is transformed into:

$$\widetilde{\mathbf{h}}'_j = \Big\Vert_{k=1}^{K} \sum_{i \in \overrightarrow{N_j}} a_{ij}^k \mathbf{m}_{ij}^k \tag{4}$$

where K denotes the number of attention heads. The attention heads perform independent self-attention over neighborhood features, then are concatenated to form the new node feature $\widetilde{\mathbf{h}}'_j$.

Another crucial step in Incoming Attention Flow is to fuse entity features with dialog context, such that even if a same set of initial entities are given, the decoder could traverse possibly different paths according to the dialog context. We achieve this fusion by concatenating the dialog context vector with entity feature computed from Eq.4 and then linear-transforming back to the entity embedding dimension:

$$\mathbf{h}'_j = \mathbf{W}_h \left[\widetilde{\mathbf{h}}'_j \,\|\, \mathbf{C}\right] \tag{5}$$

3.4 Outgoing Attention Flow

At the core of AttnIO lies the Outgoing Attention Flow, which defines path traversal on KG as an attention propagation mechanism. In the beginning of the decoding step, it starts from computing the initial attention value a_i^0 of nodes in V_{init}, i.e. the set of entities appearing in the user's last utterance.

$$a_i^0 = \underset{i \in V_{init}}{\texttt{softmax}}\left((\mathbf{W}_{init}\mathbf{C})^T \mathbf{h}_i\right) \tag{6}$$

The relevance of each candidate node is scored as the dot-product with the dialog context vector. In case of entities not in V_{init}, we initialize the node attention value to zero.

Hereafter, the decoder iterates for step 1 to T, where T denotes the maximal possible path length. We add self-loops to each node in G_{input}, in order to indicate that a traversal ended before step T. Given the context-fused entity feature $\mathbf{h}_i^t$ for all $v_i \in G_{input}$ at each step t, Outgoing Attention Flow essentially computes how much attention value to propagate from v_i to its outgoing neighbor v_j, as follows:

$$\begin{aligned} &\widetilde{a}_{ij}^{t+1} = \mathcal{T}_{ij}^{t+1} a_i^t,\ a_j^{t+1} = \sum_{i \in \overrightarrow{N_j}} \widetilde{a}_{ij}^{t+1} \\ &s.t.\ \sum_i a_i^{t+1} = 1,\ \sum_{ij} \widetilde{a}_{ij}^{t+1} = 1 \end{aligned} \tag{7}$$

A key here is the transition probability $\mathcal{T}_{ij}$, which can be derived from a function of two relevant node features, $\mathbf{h}_i$ and $\mathbf{h}_j$. In this work, we formulate the process as averaging the multi-headed attentions computed over all outgoing neighbor nodes:

$$\begin{aligned} \mathcal{T}_{ij} &= \frac{1}{K} \sum_{k=1}^{K} \underset{j \in \overleftarrow{N_i}}{\texttt{softmax}}(\tau_{ij}^k), \\ \tau_{ij}^k &= \sigma\left((\mathbf{W}_Q \mathbf{h}_i^k)^T (\mathbf{W}_k (\mathbf{h}_j^k + \mathbf{r}_{ij}^k))\right) \end{aligned} \tag{8}$$

3.4.1 Scoring candidate paths

Given the output of Outgoing Attention Flow at each decoding step i.e. the node attention distribution $a_i^0, \cdots a_i^T$ and the edge attention distribution $\widetilde{a}_{ij}^1, \cdots \widetilde{a}_{ij}^T$, we can score each candidate entity paths with the product of respective node attention value at each step:

$$\texttt{score}(P_v) = \prod_{t=0}^{T} a^t_{P_v^{(t)}} \tag{9}$$

Likewise, the score of the relation path P_r associated with P_v can be retrieved by the product of respective edge attention value at each step:

$$\texttt{score}(P_r) = \prod_{t=1}^{T} \widetilde{a}^t_{P_r^{(t)}} \tag{10}$$

3.4.2 Training Objective

We train the whole model in an end-to-end manner by directly supervising on the attention distribution at each step. In a default setting where the whole ground-truth paths are available, we use a negative log-likelihood loss on each step's attention distribution (left), and in the target-supervision setting where only the final entity labels are given, we supervise with the same loss only at the final step's attention distribution (right):

$$\mathcal{L} = \sum_t -log\, a^t_{label},\ \text{or}\ -log\, a^T_{label} \tag{11}$$

3.4.3 Dialog-KG Alignment by Initialization

AttnIO's training phase tends to be unstable in the beginning, as the model has to deal with two completely different modalities: KG entities, and the dialog. In order to align the two different features, we find that initializing each entity feature as the representation from pretrained ALBERT helps. Just as the dialogue context representation, we put each entity phrase as input with [CLS] token. We then take the hidden representation of [CLS] token from the last layer of ALBERT, and linear-transform it to create initial entity feature $\mathbf{h}_i^0$. Note that we do not fine-tune ALBERT, but back-propagate to $\mathbf{h}_i^0$ during training. This additional process not only narrows down the gap between feature space of entities and dialog contexts, but also helps better understand each entity in several cases, as some entities span a lengthy phrase of natural language tokens (e.g. *Grammy Award for Best Pop Collaboration with Vocals*).

4 Experiments and Results

Dataset We evaluate our proposed method on OpenDialKG (Moon et al., 2019), a dialog - KG parallel corpus designed for knowledge path retrieval task. The dataset consists of 91k multi-turn conversations in form of either task-oriented (recommendation) dialog, or chit-chat on a given topic. Each pair of utterances in the conversations is annotated with a KG path, where its initial entity is mentioned in the former turn, and its destination entity is mentioned in the latter turn. As the train/valid/test partitions of OpenDialKG are not publicly available, we create our own split by randomly partitioning into train (70%), valid (15%), and test set (15%).

Baselines We take 4 models suggested by Moon et al. (2019) as baselines. These models include DialKG Walker, a state-of-the-art model designed to traverse a dialogue conditioned knowledge path. Other 3 models are Seq2Seq (Sutskever et al., 2014), Extended Encoder-Decoder (Parthasarathi and Pineau, 2018), Tri-LSTM (Young et al., 2018) all modified to fit the entity path retrieval task.

Model	Recall@k									
	path@1	*path@3*	*path@5*	*path@10*	*path@25*	*tgt@1*	*tgt@3*	*tgt@5*	*tgt@10*	*tgt@25*
Seq2Seq	3.1	18.3	29.7	44.1	60.2	-	-	-	-	-
Tri-LSTM	3.2	14.2	22.6	36.3	56.2	-	-	-	-	-
EXT-ED	1.9	5.8	9.0	13.3	19.0	-	-	-	-	-
DialKG Walker	13.2	26.1	35.3	47.9	62.2	-	-	-	-	-
Seq2Path	14.92	24.95	31.1	38.68	48.15	15.65	27.04	33.86	42.52	53.28
AttnFlow	17.37	24.84	30.68	39.48	51.4	18.97	36.23	45.48	58.84	71.35
AttnIO (GRU)	22.36	36.72	42.98	51.22	61.91	23.45	42.31	51.71	64.33	77,64
AttnIO (no context)	7.27	25.12	31.03	40.39	54.72	14.33	33.3	43.26	58.32	76.49
AttnIO (no alignment)	21.84	35.19	41.19	48.85	59.08	22.99	41.3	50.63	64.01	78.02
AttnIO-AS	**23.72**	**37.53**	**43.57**	**52.17**	**62.86**	**24.98**	**43.78**	**53.49**	**65.48**	**78.79**
AttnIO-TS	12.09	23.65	30.5	39.48	51.68	22.82	40.01	49.86	61.04	74.49

Table 1: **Performance** of AttnIO in OpenDialKG, in comparison with baselines and ablation models. Results of the above 4 baselines (from Seq2Seq to DialKG Walker) are directly taken from Moon et al. (2019), as their code or implementation details are not available. Our model trained with all path supervision (AttnIO-AS) significantly outperforms all baselines.

Also, we implement another baseline model named Seq2Path, by modifying attention based Seq2Seq model to decode entity paths. On the contrary with Seq2Seq baseline in Moon et al. (2019) which added zero-shot learning layer on KG embedding as decoder, Seq2Path explicitly traverses along the graph structure by masking unreachable nodes at each decoding step. Lastly, in order to find the importance of neighbor node encoding over each entity, we suggest AttnFlow, where Incoming Attention Flow is excluded (hence node features are not updated at each step) and the Outgoing Attention Flow directly generates knowledge path from dialog context and initial entity features.

Implementation Details Our model depends heavily on message passing scheme of graph neural networks, which may lead to excessive memory usage when G_{input} is large. To further scale AttnIO to larger graphs, we reduce the size of the input graph through edge-sampling on G_{input} during training. Detailed explanation on this edge-sampling is presented in Appendix A.

As all ground-truth paths in OpenDialKG are either 1-hop or 2-hop, we set the maximal path length T = 2. We search for the best set of hyperparameters using grid-based search, choosing value with the best path accuracy with all other hyperparameters fixed. We implemented our model using PyTorch (Paszke et al., 2019) and DGL (Wang et al., 2019). Additional implementation details including hyperparameter search bounds and the best configuartion are provided in Appendix E.

4.1 Results

Table 1 presents the overall evaluation results of AttnIO, and its comparison to baseline models. In addition to the recall@k of ground-truth paths (*path@k*), we report recall@k on the target nodes (*tgt@k*), as the destination node can be considered as the most important component in knowledge path to generate response.

As can be seen in the table, our model outperforms all baselines in both *path@k* and *tgt@k*, when supervised with all entities in each path as label (AttnIO-AS). Especially, AttnIO-AS shows significantly better performance in metrics with small k. We also report our model in a more challenging setting of target supervision, assuming that only the destination node of each path is available (AttnIO-TS). In this case, our model shows a comparable target prediction performance (*tgt@k*) to AttnIO-AS, while its *path@k* is relatively poor in small ks.

Recurrent decoder based models, such as DialKG Walker and Seq2Path, relies only on a single state vector to model the transition between each decoding step. Therefore, once the model chooses to traverse a sub-optimal entity, it is hard to get back onto the right track without help of an aggressive beam search. In our method, on the contrary, the state of the decoder is essentially distributed into all the walkable entities' feature vectors; therefore, the transition is modeled alongside all the entities with nonzero attention value at each step, making the model more robust to 'misleading' hops. Also, note that AttnFlow shows consistent performance drop of about 30% then AttnIO-AS in all metrics, indicating the importance of neighborhood encod-

Dialog	A: *Fiona Stafford wrote Emma. It's a romance novel. Are you into that genre?* B: *Any other books that might fall under comedy? I'm in the mood for something light.* A: *[RESPONSE]*
AttnIO-AS	*Comedy ⇒ subject of ⇒ The War of the Worlds ⇒ written by ⇒ Arthur. C. Clarke*
AttnIO-TS	*Comedy ⇒ subject of ⇒ The War of the Worlds ⇒ subject ⇒ Comedy*
AttnFlow	*Comedy ⇒ parent genre ⇒ Slapstick*
GT	*Comedy ⇒ subject of ⇒ One Crazy Summer*

Table 2: **Sample paths** generated from each model, along with the ground-truth path. More examples are provided in Appendix D.

Model	Relation Path Accuracy
AttnIO-AS	0.403
AttnIO-TS	0.365

Table 3: **Relation Path Accuracy** at all path supervision (AttnIO-AS), and target supervision setting (AttnIO-TS).

ing step for knowledge path retrieval.

Ablation Study We conduct ablation study with three different configurations. First, we put GRU (Cho et al., 2014) as dialog encoder in replace of ALBERT, for a fairer comparison with baseline models. As shown in Table 1, we find that although the performance of AttnIO with GRU slightly degrades from that with ALBERT, it still outperforms all existing models. Next, in order to find out the value of dialog context in the traversal, we train our model with only the initial entities given as input (with uniform attention prior assigned to each initial entity), but not the dialog context. Recall@1 significantly drops in this case, while metrics with large k relatively stays moderately. This implies that although information on initial nodes appearing in last utterance might be sufficient to prune improbable paths, the dialog context is essential in finding an optimal path among probable ones. We also find in the third ablation model where no dialog-KG alignment is applied (Section 3.4.3), that ALBERT initialization of node embedding helps, leading to performance gain of about 2% in *path@1*.

4.2 Analysis

Relation Accuracy The poor entity path accuracy of AttnIO-TS may seem natural, as the initial node and intermediate node (in case of multi-hop) are not given as label in target supervision setting. However, one should note that there can be a variety of entity paths that match human sense in naturalness and coherence for a specific dialog. For an example shown in Table 2, any film of comedy genre shall replace *One Crazy Summer* in GT-path, without loss of naturalness. The generated path from AttnIO-AS could even be an answer, giving more information on the chosen film. The inherent one-to-many relationship between dialog context and probable knowledge, makes it hard to correctly assess the performance of knowledge retrieval models. Relation path accuracy could be one way to relieve this problem, as relations represent important attributes shared by similar entities.

The relation path accuracy of AttnIO in both supervision setting is as shown in Table 3. The relation path accuracy under both setting is clearly higher than the entity path accuracy, implying the generalization capability of our model based on reasoning over relations, rather than depending on specific entities. Notably, AttnIO-TS shows only about 10% relative difference from AttnIO-AS, unlike in entity *path@1* in Table 1. This indicates that our model can learn to competently perform relational reasoning, even in this in-the-wild setting of target supervision.

	vs. GT		
Model	Win	Tie	Lose
AttnIO-AS	11.2%	55.2%	33.6%
AttnIO-TS	17.6%	55%	27.4%

Table 4: **Pairwise human evaluation** results between model-generated paths, and ground-truth paths.

Human Evaluation In order to further examine the quality of paths from the two supervision setting, we conduct a human evaluation. We randomly sample 100 dialogues from test set, then generate knowledge paths for half of the dialogues from AttnIO-AS, and half of the dialogues from AttnIO-TS. We then perform a pairwise comparison between the path generated from AttnIO, and the ground-truth path actually used in the dataset. For each dialogue, we ask 5 crowd-source workers to evaluate which knowledge path is more suitable for response generation among the two.

We report the win/tie/lose statistics of the model generated paths against ground-truth paths in Table 4. In both all-path supervision and target supervision setting, more than half of the paths from our model tied with the actual paths. The result attests to the quality of the generated paths, even including those marked as wrong in quantitative measures.

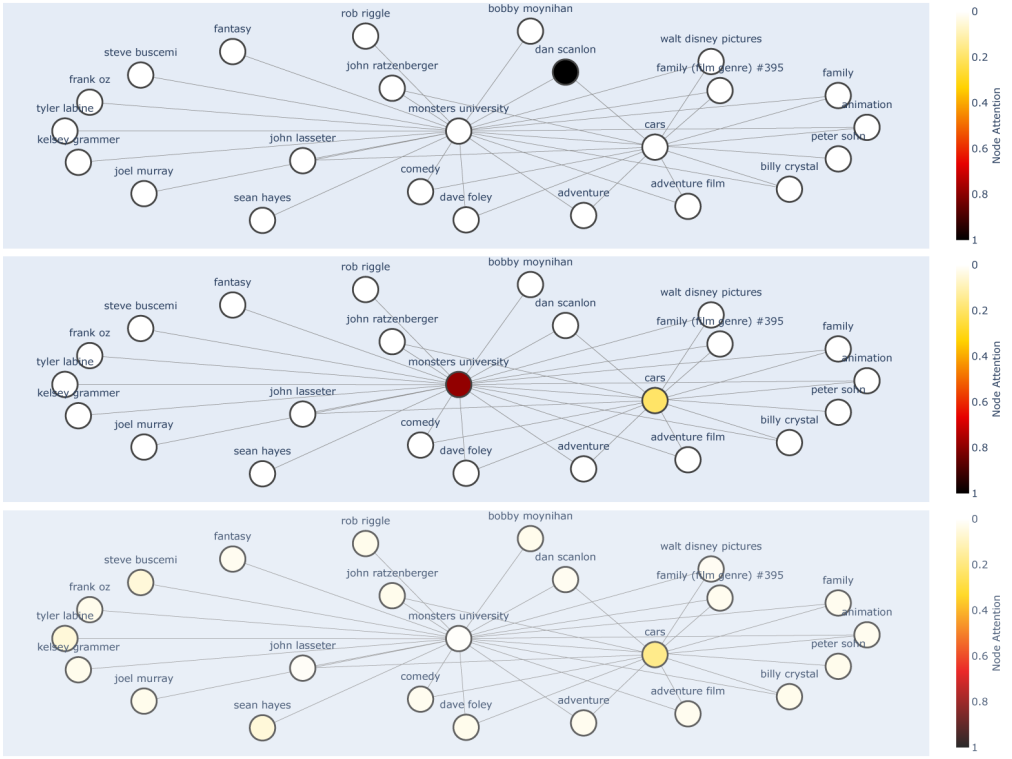

Figure 3: **Node attention visualization** from the case study. Each figure represents the node attention at the initial state (Top), after the first decoding step (Center), and after the second decoding step (Bottom). We omit the edge attention to avoid visual cluttering. Best viewed in color.

AttnIO-TS especially performs much more comparably to AttnIO-AS than in Table 1, indicating that the destination nodes can function as an adequate guidance to our model, in replace of the whole path label.

Case Study We resort to a case study, for a clear presentation of AttnIO's path reasoning process. Figure 3 presents the visualization of output attention distribution from our model, when the dialog context is given as follows:

> A: *Can you recommend some films by Dan Scanlon?*
> B: *[RESPONSE]*

Note that there are hundreds of neighbor nodes connected to each entity in the external KG, but for the sake of clarity, we pruned most of them in the visualization leaving only entities relevant to the dialog. Intuitively, there could be diverse knowledge paths as response to the user's question. Before the initial step, AttnIO starts from assigning an attention value of 1.0 to the only entity mentioned in the utterance, *Dan Scanlon*. In the first propagation step, our model finds from the dialog context, that the most relevant relation in this case is *wrote*, propagating most attention in *Dan Scanlon* to two movie entities, *Monster's University* and *Cars*. In the second step, AttnIO understands that most of the entities directly connected to these two movies, can be a good option for the destination node. As a consequence, AttnIO chooses to propagate a fair amount of attention value evenly to all reachable entities, resulting in the distribution visualized at the third figure. Finally, an optimal path can be retrieved as *Dan Scanlon* ⇒ *wrote* ⇒ *Monster's University* ⇒ *starred actor* ⇒ *Steve Buscemi*.

Through the case study, we find that AttnIO directly reflects human intuition regarding an open question. It learns to perform relation-centric reasoning, and assign even amount of attention to equally likely reachable entities. In contrast, given a closed question such as "Who direced movie *Cars*?", AttnIO focuses on a small set of relevant entities and relations. A detailed analysis on the contrasting example is provided in Appendix C.

5 Conclusion

In this work, we suggest AttnIO, a novel path traversal model that reasons over KG based on two directions of attention flows. The empirical evaluations

on OpenDialKG dataset show the strength of AttnIO in knowledge retrieval compared to baselines. AttnIO can also be trained to generate proper paths even in a more affordable setting of target supervision. Lastly, we show through case study that our model enjoys from transparent interpretation of path reasoning process, and is capable of intuitively modeling knowledge exploration depending on the dialog characteristics.

References

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

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1803–1813.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *6th International Conference on Learning Representations, ICLR 2018*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A Smith. 2019. Show your work: Improved reporting of experimental results. *arXiv preprint arXiv:1909.03004*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253, Brussels, Belgium. Association for Computational Linguistics.

Shuman Liu, Hongshen Chen, Zhaochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1498. Association for Computational Linguistics.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019. Knowledge aware conversation generation with explainable reasoning over augmented graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1782–1792, Hong Kong, China. Association for Computational Linguistics.

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.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4710–4723.

Tu Dinh Nguyen, Dat Quoc Nguyen, Dinh Phung, et al. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 327–333.

Prasanna Parthasarathi and Joelle Pineau. 2018. Extending neural generative conversational model using external knowledge sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 690–695, Brussels, Belgium. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3060–3067.

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

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning (ICML)*. International Conference on Machine Learning (ICML).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, Ziyue Huang, Qipeng Guo, Hao Zhang, Haibin Lin, Junbo Zhao, Jinyang Li, Alexander J Smola, and Zheng Zhang. 2019. Deep graph library: Towards efficient and scalable deep learning on graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Xiaoran Xu, Songpeng Zu, Chengliang Gao, Yuan Zhang, and Wei Feng. 2018. Modeling attention flow on graphs. *arXiv preprint arXiv:1811.00497*.

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. 2018. Augmenting end-to-end dialogue systems with commonsense knowledge. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2019. Conversation generation with concept flow. *arXiv preprint arXiv:1911.02707*.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

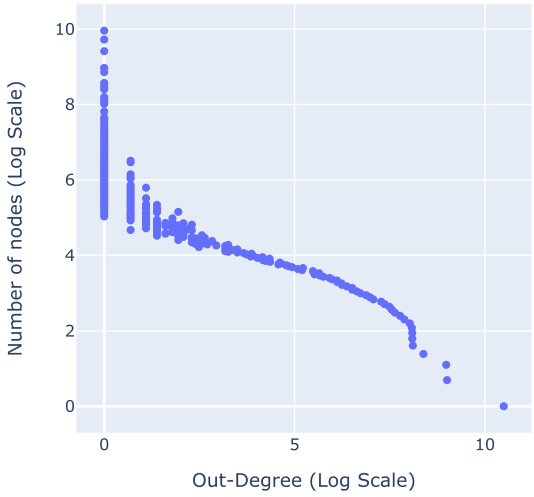

Figure 4: **Out-degree distribution** of all nodes in G_{KG}. Both axis are in log scale.

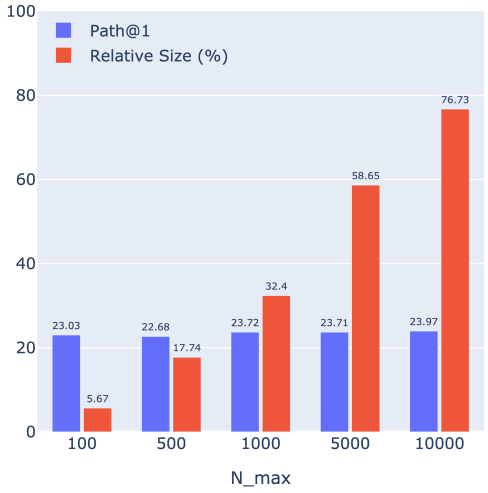

Figure 5: **Effect of subgraph sampling**. Blue bar denotes *path@1* for each N_{max}, while red bar denotes the average relative size of the sampled subgraph compared to the original G_{input}.

A Subgraph Sampling

We explain in more detail the subgraph sampling method adopted by AttnIO, as mentioned in Section 4.

As shown in Figure 4, the out-degree distribution of G_{KG} follows an extreme power-law distribution, which is typical in relational graphs. Among 100K nodes in G_{KG}, about 31K nodes possess only one incoming neighbor node, making the graph extremely sparse. Meanwhile, a node with the highest in-degree has more than 21K incoming neighbor nodes, connecting the node to about 20% of all entities in the whole graph.

We find that the small number of *hub nodes* with high in/out degrees are the major factor that increases the size of input graph G_{input}. Therefore, we choose to limit the maximal number of neighbors to sample from each entity, while constructing G_{input} in the training time. We denote this limit as N_{max}.

The effect of subgraph sampling with different N_{max} is shown in Figure 5. Setting N_{max} to 100, subgraph sampling effectively reduces down the number of edges in the input graph to only 5.67% of the original G_{input} on average, while losing only about 1.0 absolute performance in *path@1*. In all our experiments, we set $N_{max} = 1000$, leaving only 32.4% of the edges originally in G_{input}, while not compromising for the retrieval accuracy.

B Dataset Statistics

Dialog		KG	
# of dialogues:	15,673	$\|V\|$:	100,813
# of turns:	91,209	$\|E\|$:	1,190,658

Table 5: **Dataset Statistics** of OpenDialKG.

The statistics of OpenDialKG dataset is as shown in Table 5. There are 1358 distinct types of relations comprising 1M+ edges.

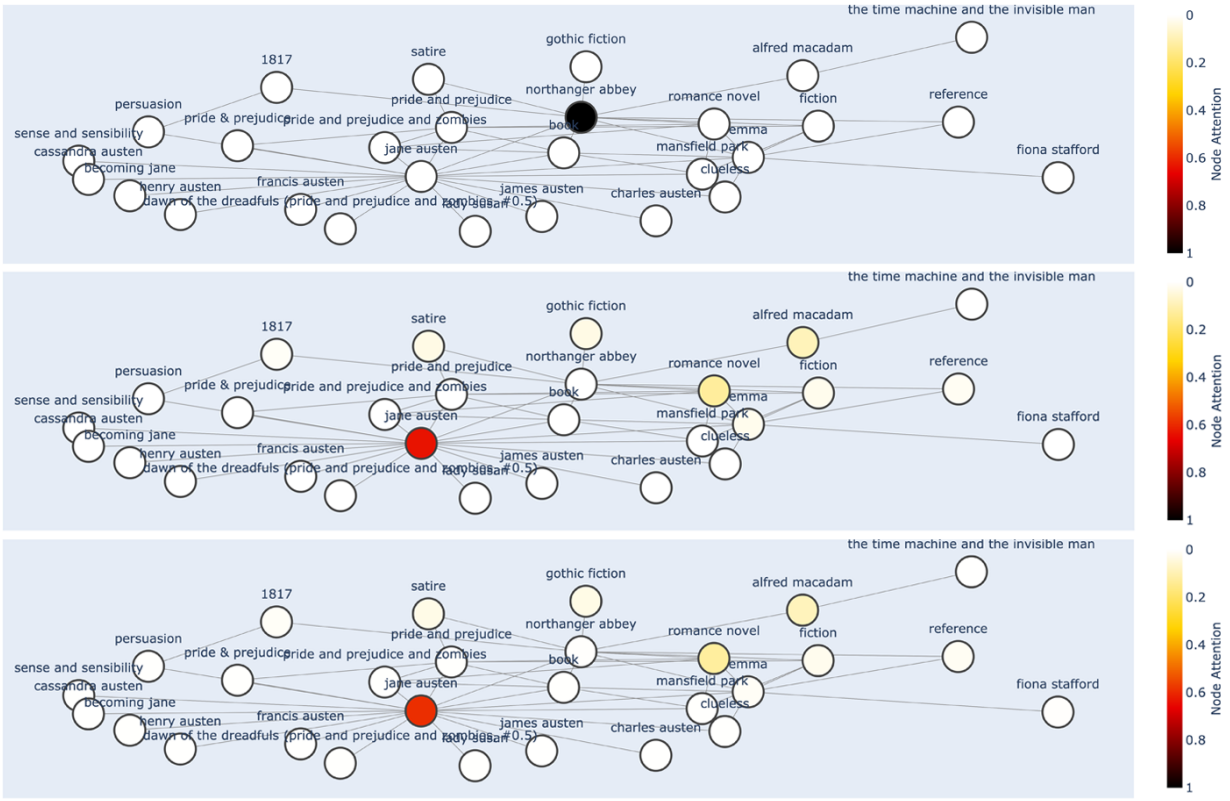

Figure 6: **Node attention visualization** for the additional case study. Each figure represents the node attention at the initial state (Top), after the first decoding step (Center), and after the second decoding step (Bottom). The center and bottom figure might look similar, due to the slight difference of node attention distribution between the two steps. The attention is focused on small number of entities, on the contrary with the even distribution at the former case study.

C Additional Case Study

We provide an additional visualization of node attention distribution for a given dialogue in Figure 6. This time, the model is given a dialogue context as following:

> A: *Someone suggested the Northanger Abbey book to me. Do you know who the author is?*
> B: *[RESPONSE]*

The dialogue is of a similar topic with the case study provided in Section 4.2, but the query in this case is a closed question that specifically asks for the author of a book.

AttnIO first starts from the only initial entity *Northanger Abbey*, and finds from the context that the most relevant relation here is *written by*. Therefore in the first decoding step, AttnIO propagates more than half of the attention value (0.63) from *Northanger Abbey* to its writer, *Jane Austen*. In the second propagation step, AttnIO chooses not to propagate much attention to any of *Jane Austen*'s neighbor nodes, preserving most of the attention value (0.59) by traversing a self-loop. (We omitted self-loops in the visualization for clarity.)

Note that there are a variety of neighbor nodes walkable from *Jane Austen*, just as in the former case study. However, AttnIO understands the intent of user's utterance requiring for a specific answer, concentrating most of the attention to the nodes and edges directly related to the dialogue context.

D Generation Examples

In Table 6, we present more path generation examples along with ground-truth paths for the given dialogues. Note that we only sampled cases where the paths generated from our model are different from the ground-truth paths. Dialogs are partially shown to meet the spatial constraints.

Dialog	A: *I'm not sure who else was in it, but Ralph Fiennes also starred in Wrath of the Titans.* B: *Wrath of the Titans, I didn't know Ralph Fiennes was in that movie. Tell me more about that movie and the stars in it.* A: *[RESPONSE]*
MODEL-AS	*Wrath of the Titans ⇒ starred ⇒ Liam Neeson*
MODEL-TS	*Wrath of the Titans ⇒ starred ⇒ Liam Neeson*
AttnFlow	*Ralph Fiennes ⇒ starred ⇒ The hurt Locker*
GT	*Ralph Fiennes ⇒ starred ⇒ Wrath of the Titans ⇒ written by ⇒ Greg Berlanti*

Dialog	A: *I think Tiger Woods is a good golf player, but is he retired right now?* B: *No he is actually still playing. Is he half asian?* A: *[RESPONSE]*
MODEL-AS	*Asian ⇒ ethnicity of ⇒ Tiger Woods*
MODEL-TS	*Asian ⇒ ethnicity of ⇒ Tiger Woods*
AttnFlow	*Asian ⇒ language ⇒ Vietnamese Language*
GT	*Asian ⇒ includes ⇒ Vietnamese American*

Dialog	A: *Could you recommend books written by Aldous Huxley?* B: *[RESPONSE]*
MODEL-AS	*Aldous Huxley ⇒ wrote ⇒ The doors of perception & heaven and hell*
MODEL-TS	*Aldous Huxley ⇒ wrote ⇒ Brave new world*
AttnFlow	*Aldous Huxley ⇒ cause of death ⇒ Laryngeal Cancer*
GT	*Aldous Huxley ⇒ wrote ⇒ Island*

Dialog	A: *Drew Brees is a quarterback for the new orleans saints. I don't follow football but I hear he is pretty good.* B: *I like movies more than football. I actually liked the american football movies.* A: *[RESPONSE]*
MODEL-AS	*American Football ⇒ subject of ⇒ Wild Cats ⇒ starred actor ⇒ Goldie Hawn*
MODEL-TS	*American Football ⇒ subject of ⇒ Wild Cats ⇒ has genre ⇒ Football*
AttnFlow	*American Football ⇒ sports played ⇒ Troy Aikman*
GT	*American Football ⇒ subject of ⇒ Rudy ⇒ has genre ⇒ Football*

Table 6: **Generated path examples**, along with the ground-truth paths.

E Additional Implementation Detail

Computing Infrastructure	Tesla V100 GPU
Search Strategy	Manual Tuning
Best Validation *path@1*	23.72 (AS), 12.18 (TS)
Training Time (per epoch)	$\approx$ 64min

Hyperparameter	**Search Bound**	**Best Setting**
max path length T	2	2
subgraph sampling limit N_{max}	*choice*[100, 500, 1000, 5000, 10000]	1000
max dialog history	*choice*[3, 4, 5, 6]	3
entity feature dimension	*choice*[60, 80, 100, 120]	80
number of attention heads	*choice*[3, 4, 5, 6]	5
number of epochs	20	20
batch size	*choice*[4, 8, 16]	8
optimizer	*Adam*	*Adam*
learning rate	*loguniform-float*[5e-2, 5e-5]	5e-4
lr scheduler	*reduce_on_plateau*	*reduce_on_plateau*
lr reduction factor	0.1	0.1
gradient clip norm	*uniform-integer[3, 10]*	5

Table 7: **Additional implementation detail** of AttnIO. We follow the specification from Dodge et al. (2019) by reporting hyperparameter search spaces and experimental details.