Augmenting Topic Aware Knowledge-Grounded Conversations with Dynamic Built Knowledge Graphs

Junjie Wu[♠,♣] **Hao Zhou**[♢]
[♠]AI Thrust, Information Hub, HKUST
[♣]Department of Computer Science and Technology, HKUST
[♢]Department of Computer Science, Tsinghua University
junjie.wu@connect.ust.hk
zhouhao16@mails.tsinghua.edu.cn

Abstract

Dialog topic management and background knowledge selection are essential factors for the success of knowledge-grounded open-domain conversations. However, existing models are primarily performed with symmetric knowledge bases or stylized with pre-defined roles between conversational partners, while people usually have their own knowledge before a real chit-chat. To address this problem, we propose a dynamic knowledge graph-based topical conversation model (DKGT). Given a dialog history context, our model first builds knowledge graphs from the context as an imitation of human's ability to form logical relationships between known and unknown topics during a conversation. This logical information will be fed into a topic predictor to promote topic management, then facilitate background knowledge selection and response generation. To the best of our knowledge, this is the first attempt to dynamically form knowledge graphs between chatting topics to assist dialog topic management during a conversation. Experimental results manifest that our model can properly schedule conversational topics and pick suitable knowledge to generate informative responses comparing to several strong baselines.

1 Introduction

Conversational AI, especially the open-domain dialog system, is an essential and challenging problem that leads to a variety of applications (Vinyals and Le, 2015; Serban et al., 2017). Previous works introduce external background knowledge to help their systems generate more informative responses (Li et al., 2016b; Dinan et al., 2018; Ghazvininejad et al., 2018; Young et al., 2018). However, these systems are facing a main issue that they tend to only utilize dialog utterances as queries to match appropriate knowledge sentences. Table 1 shows two responses corresponding to the same post. As can be seen, response1 changes the chatting topic abruptly and thus becomes incoherent. By contrast, response2 first manages to deepen the current topic "William Shakespeare", then picks a suitable knowledge candidate to generate an engaging response. Therefore, a good topic managing strategy is also very crucial to dialog generation.

Chatting topics: William Shakespeare; Sun; Jane Austen **Knowledge:** ..... Shakespeare invented the names Miranda, Jessica, and Olivia. .....
Dialog ..... **A**: Do you like shakespeare? **B**: Yes a little bit. He is often called england' s national poet and the "bard of avon". **A**: He is a great dramatist that influenced a lot of people, like Joenesbo.
Response 1: Did you know that Ronald Reagan was rejected for a movie role because an entertainment executive didn' t look presidential enough? **Response 2:** I love shakespeare' s works. Did you know that he invented the names Miranda, Jessica, and Olivia ?

Table 1: Example responses generated by two models.

To solve this problem, some papers propose to plan a set of conversational topics as chatting goals in advance to boost knowledge matching and response generation (Wu et al., 2019; Xu et al., 2020). However, it is difficult to schedule an appropriate topic transition route beforehand since topics are switching dynamically during a chit-chat based on many real-time factors, especially when two partners have different personal knowledge. Hence, these methods could not pre-schedule a topic at each turn properly and thus becoming non-attractive.

Another problem these knowledge-grounded or topic-enhanced models might encounter is that they are typically tested under symmetric knowledge settings (Young et al., 2018), or asymmetric settings with pre-defined roles (Dinan et al., 2018). Yet people usually have unequal personal knowledge prior to real-world conversations. Hence, such models cannot reflect the effect of information transferring

Proceedings of Deep Learning Inside Out (DeeLIO):
The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 31–39

and learning between two strangers, which is crucial to an engaging conversation. This issue will matter more when there are no pre-defined roles between two conversation partners.

To solve these problems, in this paper, we study the problem of topic transitions in open-domain conversations under both symmetric and asymmetric settings. To this end, we propose a dynamic knowledge graph-based topical conversation model (DKGT). Given a dialogue context and a corresponding knowledge base, we first extract knowledge triples from each utterance and then jointly combine those triples through a static graph attention mechanism. Such logical information will then be fed into a topic predictor to predict the next chatting topic, which assists background knowledge selection and dialog generation. We further demonstrate the effectiveness of our method on Topical-Chat (Gopalakrishnan et al., 2019), comparing to several baselines. The main contributions of this paper can be wrapped as follows:

- To the best of our knowledge, this is the first attempt to dynamically mine logical relationships between chatting topics during a conversation to assist topic management, in the form of knowledge graphs.
- The proposed model has two benefits: 1. The dynamic built KG can automatically form logical information between chatting topics during a conversation, which helps our system to learn from its conversational partner especially when they have different prior knowledge. 2. Such logical information can be used to facilitate topic transition and background knowledge selection, then prompts coherent dialog generation.
- Experimental results demonstrate that our method is capable of scheduling appropriate topics and picking suitable background knowledge to generate informative and diverse responses.

2 Related Work

Knowledge-Grounded Open-Domain Dialog Systems Since traditional end-to-end architectures (Li et al., 2015; Serban et al., 2017) often generate generic and dull responses, several works introduce external background knowledge to produce diverse context (Ghazvininejad et al., 2018; Dinan et al., 2018; Li et al., 2019). Although these methods have obtained promising results, they are facing two main issues. First, such models are agnostic to internal topic coherence, which usually leads to less logical conversations. Second, their conversation partners often have pre-defined roles or the external knowledge provided for these partners is usually symmetric, which could not reflect real-world chit-chats.

Topic-aware Conversational models A variety of approaches proposed to leverage topic information by recognizing topic words inside or outside the previous utterances (Xing et al., 2017; Wang et al., 2018; Dziri et al., 2019). However, simply fusing topic words into text representations makes these models ignore logical relationships between topics and thus fail to perform smooth topic transitions. Except for applying attention mechanism on topic words, researchers have also investigated proactive conversation, whose topic transition and conversation development are conditioned on pre-scheduled chatting goals (Li et al., 2018; Wu et al., 2019; Xu et al., 2020). Nevertheless, these models are limited by pre-defined topic and goal sequences, hence not applicable to open-domain and open-topic conversations.

Graph-enhanced Conversational Models Structured knowledge has been studied to improve dialog generation for a long time (Hixon et al., 2015; Zhou et al., 2018; Moon et al., 2019; Tuan et al., 2019). However, these models are mainly based on pre-defined knowledge graphs, which restrict their ability under symmetric knowledge settings. By contrast, our dynamic knowledge graphs enable our system to learn logical information through the conversation like humans, which facilitates both topic management and dialog generation.

3 Model

3.1 Task Definition and Overview

Formally, let $D = [x_1, x_2, ..., x_k]$ be a conversation history including k utterances, where $x_i (1 \leq i \leq k)$ stands for the i^{th} utterance. Each x_i is associated with its speaker's background knowledge set K, which has been segmented into several sentences $[k_1, k_2, ..., k_j]$. Given such information, our goal is to predict the chatting topic at each turn and make good use of knowledge set K to generate a coherent response x_{i+1}.

Figure 1 shows the overview architecture of our proposed DKGT. Given a conversation history D,

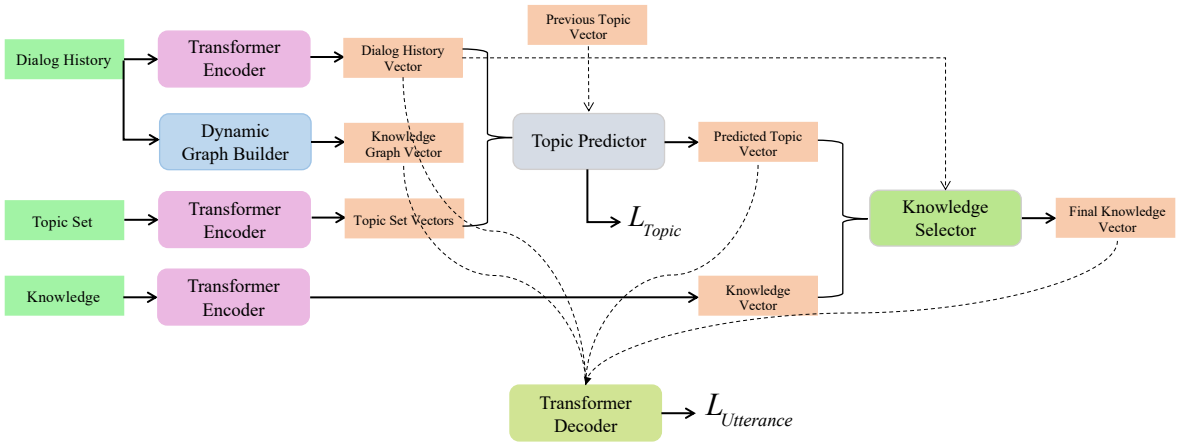

Figure 1: Overview of our proposed DKGT model.

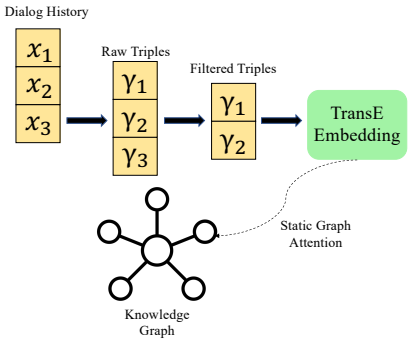

Figure 2: Structure of the Dynamic Graph Builder.

the dynamic graph builder first extracts knowledge triples across specified entities to construct a knowledge graph $G = [\gamma_1, \gamma_2, ..., \gamma_m]$, where each triple is denoted as $\gamma = (h, r, t)$ (head entity, relation, tail entity). We then adopt TransE (Bordes et al., 2013) to obtain vector representation $\boldsymbol{G}$ of knowledge graph G. Next, the topic predictor takes encoded representation $\boldsymbol{D}$ and $\boldsymbol{G}$ as input and applies a MLP-based module to predict the topic label T. The assigned T enables our model to decrease knowledge set K to a smaller one $\boldsymbol{K}$ and thus facilitates further knowledge acquisition. Afterward, the knowledge retriever adopts another attention mechanism to obtain a cumulative knowledge representation $\boldsymbol{K}$ of the decreased knowledge set K. Finally, dialog context $\boldsymbol{D}$, accumulated knowledge $\boldsymbol{K}$, topic vector $\boldsymbol{T}$ and graph vector $\boldsymbol{G}$ are concatenated orderly and fed into a transformer decoder. Our transformer decoder will then attentively read the concatenated vector and generates an informative response.

3.2 Dynamic Graph Builder

As shown in Figure 2, at each turn, the model first extracts knowledge triples from all individual sentences that are longer than three words using an open-source relation extraction tool OpenNRE (Han et al., 2019) [1]. With preassigned entities, OpenNRE can provide a relation between the head and tail entity along with a confidential probability score. In this work, the scope of graph entities is limited to two categories: topic entities provided by the Topical-Chat dataset and named entities (except numerical entities like date and time). To confirm the quality of generated triples, we manually set 0.7 as a threshold probability score to perform filtering. When a new utterance x_{i+1} comes in, the knowledge graph G will be updated dynamically following the above procedure. This strategy enables our system to learn knowledge and form new logical relationships in real-time.

For each triple $\gamma = (h, r, t)$ in graph G, we adopt TransE (Bordes et al., 2013) to obtain its low-dimension representation. Moreover, to fill in the representation gap between raw utterances and triples, we employ an MLP layer on those triple embeddings. Therefore, a triple vector $\boldsymbol{\gamma}$ can be further denoted as $\boldsymbol{\gamma} = (\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$, where $\boldsymbol{h}$, $\boldsymbol{r}$, $\boldsymbol{t}$ are the transformed TransE embeddings of h, r, t respectively.

The static graph attention mechanism (Zhou et al., 2018) is then applied to capture semantic information from entities and their relations. All the knowledge triple vectors $[\boldsymbol{\gamma_1}, \boldsymbol{\gamma_2}, ..., \boldsymbol{\gamma_m}]$ are jointly computed to generate a graph vector $\boldsymbol{G}$:

[1] https://github.com/thunlp/OpenNRE

$$\mu_i = (\boldsymbol{W_r r_i})^\top tanh(\boldsymbol{W_h h_i} + \boldsymbol{W_t t_i}), \quad (1)$$

$$\eta_i = \frac{exp(\mu_i)}{\sum_{j=1}^{m} exp(\mu_j)}, \quad (2)$$

$$\boldsymbol{G} = \sum_{i=1}^{m} \eta_i [\boldsymbol{h_i}; \boldsymbol{t_i}], \quad (3)$$

where $\boldsymbol{W_h}$, $\boldsymbol{W_r}$, $\boldsymbol{W_t}$ are weight matrices for $\boldsymbol{h}$, $\boldsymbol{r}$, $\boldsymbol{t}$, and $[\boldsymbol{h_i}; \boldsymbol{t_i}]$ denotes the concatenated vector of $\boldsymbol{h_i}$ and $\boldsymbol{t_i}$.

3.3 Encoder

A shared transformer-based (Vaswani et al., 2017) encoder is employed to encode dialog utterances, knowledge sentences and topic labels. In this task, the topic label T_i of each utterance x_i is a word or phrase that refers to an entity in the pre-defined topic entity set [2]. To better capture the structural information between utterances, the dialog history input at turn i is defined as the concatenation of previous utterances $D_i = [x_1; x_2; ...; x_i]$. We use $\boldsymbol{D_i}, \boldsymbol{k_i}, \boldsymbol{T_i}, \boldsymbol{T_s}$ to symbol the encoded counterparts, where $\boldsymbol{T_s}$ is the encoded representation for all topic entities in the pre-defined set.

3.4 Topic Predictor

At each turn i, upon obtaining knowledge graph vector $\boldsymbol{G}$, dialog history vector $\boldsymbol{D_i}$,topic vector $\boldsymbol{T_i}$ and $\boldsymbol{T_s}$, a three-layer MLP-based module is applied to predict the topic label of the next utterance. We concatenate the first token's (the [CLS] token) hidden state from both dialog context vector $\boldsymbol{D_i}$, topic vectors $\boldsymbol{T_i}$ and $\boldsymbol{T_s}$ as input to attain a probability distribution of the next topic label T_{i+1}:

$$T_{i+1} = Softmax(\boldsymbol{MLP}([\boldsymbol{D_{i_{first}}}; \boldsymbol{T_{s_{first}}}; \boldsymbol{T_{i_{first}}}; \boldsymbol{G}])), \quad (4)$$

where $\boldsymbol{D_{i_{first}}}$, $\boldsymbol{T_{s_{first}}}$ and $\boldsymbol{T_{i_{first}}}$ are the first token's hidden states for $\boldsymbol{D_i}$,$\boldsymbol{T_i}$ and $\boldsymbol{T_s}$.

3.5 Knowledge Retriever

During the whole chat, each speaker has access to a specific knowledge set K that includes dozens of candidates k_i. However, it is challenging for existing models to handle large knowledge bases at once. Hence, we operate a general attention mechanism between dialog context and knowledge candidates to get an informative knowledge representation. A sentence embedding layer (Cer et al., 2018) will first obtain sentence-level representations of $\boldsymbol{D_i}$ and knowledge representation $\boldsymbol{k_i}$ as $\boldsymbol{D_i^s}$ and $\boldsymbol{k_i^s}$. Next, given the predicted topic T_{i+1}, we can pick T_{i+1} related knowledge from the original knowledge set to form K_{small}. Our model then orderly attends on each knowledge candidate in K_{small} to generate a knowledge representation for the next turn as below:

$$\alpha_m = \boldsymbol{k_m^{s\top}} \boldsymbol{W_{D_i^s}} \boldsymbol{D_i^s}, \quad (5)$$

$$\beta_m = \frac{exp(\alpha_m)}{\sum_{j=1}^{N_{K_{small}}} exp(\alpha_j)}, \quad (6)$$

$$\boldsymbol{K_{i+1}} = \sum_{m=1}^{N_{K_{small}}} \beta_m \boldsymbol{k_m^s}, \quad (7)$$

where $\boldsymbol{W_{D_i^s}}$ is the weight matrix for $\boldsymbol{D_i^s}$ and $N_{K_{small}}$ is the number of candidates in K_{small}. $\boldsymbol{K_{i+1}}$ is the final knowledge representation to be used in the decoding part.

3.6 Decoder and Loss Function

As illustrated in Figure 1, we adopt another transformer as a decoder to generate coherent responses, whose structure is the same as the encoder. Our decoder generates responses with the following procedure:

Formally, let the gold token distribution be Γ_k and the predicted token distribution be Δ_k, we optimize our model's parameters by minimizing the cross entropy error between these two distributions. Besides, we employ supervised signals on the topic predictor to teacher-force the model to predict a suitable topic. Finally, the loss function between generated sequence $\boldsymbol{Y}$ and ground truth $\boldsymbol{X}$ ($\boldsymbol{X} = (x_1, x_2, ..., x_n)$, $\boldsymbol{Y} = (y_1, y_2, ..., y_m)$) at turn i is formulated as:

$$L(\theta) = -\lambda_1 \sum_{k=1}^{m} \Gamma_k log(\Delta_k) - \lambda_2 \sum_{j=1}^{N_{set}} T^g_{i+1_j} log(\boldsymbol{T^p_{i+1_j}}), \quad (8)$$

where $T^g_{i+1_j}$ and $\boldsymbol{T^p_{i+1_j}}$ are the ground truth label and predicted probability distribution at turn $i+1$ respectively. λ_1 and λ_2 stand for the weights of our two loss terms, and N_{set} is the number of topic labels in the pre-defined topic set. In our experiments, λ_1 and λ_2 are set to 1 and 20.

[2] Topical-chat provides three topic entities for each conversation, which consists of our topic set.

	Config	Train	Valid Freq	Valid Rare	Test Freq	Test Rare
Dialogs	A	1181	62	73	44	70
	B	1144	54	75	34	78
	C	1298	58	78	31	72
	D	1205	54	80	122	85
	Total	4828	228	306	231	305
Utterances	A	24,609	1,272	1,531	934	1,466
	B	23,888	1,118	1,548	699	1,632
	C	27,151	1,225	1,624	647	1,488
	D	25,199	1,137	1,654	2,579	1,816
	Total	100,847	4,752	6,357	4,859	6,402

Table 2: Statistics of the dataset.

Set	Total Number	Averaged per Dialog
Train	52829	10.9
Valid Freq	2210	9.7
Valid Rare	3070	10.0
Test Freq	2139	9.3
Test Rare	3115	10.2

Table 3: Statistics of extracted triples on different sets.

4 Experiments

4.1 Dataset

In this work, we use a public released dataset Topical-Chat [3] , which contains thousands of knowledge-grounded conversations spanning 300 specific topics. [4] To enhance the effect of knowledge graphs on this dataset, we firstly used OpenNRE to extract triples for every utterance, then filtered out conversations with less than 5 triples. The statistics of our downsampled dataset and extracted triples are shown in Table 2 and Table 3 respectively.

4.2 Obtaining Topic Labels

Although Topical-Chat does not provide topic annotation for each utterance directly, workers have attached their choice of knowledge scope at each turn during crowd-sourcing, which can then be converted into topic labels automatically. Hence, we first obtained ground truth topic labels in the following steps:

1. If a given knowledge source is solely related to a fun fact under one of the topic entities, this topic entity will become the topic label.

2. If a given knowledge source is solely related to an article sentence, the topic which appears most frequently will become the topic label.

3. While an utterance equips multiple knowledge sources, we take its closest utterance(e.g. $i+1$ and $i-1$ given i) 's topic as the topic label.

4. If "Personal Knowledge" appears in the knowledge source, we ignore it for simplicity. When "Personal Knowledge" is the only knowledge source, the strategy in step3 will be performed to generate a topic label.

Although step3 acts as an estimate of the current chatting topic and might bring some biases, our topic annotation is still effective in two ways: **First**, the accurate annotation step1 and step2 have covered most of the utterances in the dataset (e.g. 82.3 % in the training corpus); **Second**, topic transition in a dialog usually happens after several turns, which means the topic label at turn i is often the same as turn $i-1$ and turn $i+1$. More importantly, the above strategy highly reduces human efforts when annotating data.

4.3 Baselines

To make an empirical comparison, we choose the following baseline models:

Seq2Seq-TF: A simple sequence to sequence architecture (Vinyals and Le, 2015) that applies transformer-based encoder and decoder. We also add a topic classifier at the top of each utterance representation to perform topic prediction for further comparison.

Wizard-TF is a transformer-based memory network for document-grounded open-domain dialog generation (Dinan et al., 2018). It takes context

[3] https://github.com/alexa/alexa-prize-topical-chat-dataset

[4] We recommend readers to read (Gopalakrishnan et al., 2019)

vectors as queries to match a single knowledge candidate to generate responses. A topic predictor is also added for our evaluation.

For our proposed **DKGT** model, we further devise two variants for comparison and ablation study:

DKGT w/ all Know is used to evaluate the effect of the topic predictor. After predicting a topic for the next turn, the size of the knowledge set will not be decreased by the predicted topic and the model needs to match background knowledge from the raw knowledge base.

DKGT w/o Graph: A variant without the dynamic graph module. This setup aims to check the effectiveness of our proposed dynamic graph builder.

4.4 Implementation Details

We apply PyTorch [5] to perform all the experiments. During data preprocessing, the max sequence length is 128 for dialog history utterances, 50 for responses, 64 for knowledge candidates, and 10 for topic entities respectively. For Wizard Tranformer, we follow their original hyperparameter settings. For other transformer-based models, their hidden size is 512. The number of layers of encoder and decoder is set to 3 while the number of attention heads in multi-head attention is 4. All the transformer modules are based on Hugging Face's framework [6]. In the dynamic graph builder, we adopt TransE (Bordes et al., 2013) to generate entity and relation representations, and the embedding size of both entities and relations is set to 100. For decoding, we apply the Top-p sampling strategy proposed by (Holtzman et al., 2019) with a temperature of 0.7, and the threshold of the cumulative probability is 0.9. During training, we use the AdamW optimizer with a batch size of 64. The gradient clip is limited to 0.1. We take the first two training epochs as a warm-up process and the learning rate is set to 0.0001(except Wizard-TF). All the models are trained for at most 30 epochs and the training stops when the perplexity on the validation freq set starts to increase. The training stage of each model took about two and a half days on a Titan X GPU machine.

4.5 Automatic Evaluation

Metrics: In our experiments, we use perplexity (PPL) and BLEU 1-gram to evaluate our system at the content level. We also adopt distinct 1-gram (Li et al., 2016a) to assess the diversity of generated responses. To evaluate our models at the topic level, we calculate the accuracy between the predicted topic label and the ground truth topic label.

Model	PPL	BLEU-1%	Dist-1%	Acc
Seq2Seq-TF	36.82	23.03	1.37	0.395
Wizard-TF	37.67	22.41	1.41	0.307
DKGT w/*all Know*	36.54	23.44	**1.49**	**0.782**
DKGT w/o *Graph*	**35.97**	23.41	1.42	0.765
DKGT	36.08	**23.58**	1.46	0.780

Table 4: Automatic evaluation results.

Results: Table 3 shows the automatic evaluation results for all the models. As can be seen, DKGT outperforms Wizard-TF and Seq2Seq-TF on all the metrics, demonstrating that our model can generate more fluent and informative responses with the help of all the proposed strategies. Moreover, the topic accuracy scores of Seq2Seq-TF and Wizard-TF are extremely low. This is due to their lack of additional supervision signals on topic labels during training.

To examine the influence of different modules, we also perform an ablation study using two variant models. As we can see, after removing the dynamic kg module, topic accuracy drops a lot, proving that the dynamic kg module augments our system's ability to manage dialog topics since it stores logical information between topic entities. Although DKGT *w/o Graph* attains the lowest perplexity score, our model not only achieves a similar perplexity score but also obtains the highest BLEU-1 value, showing that it can perform proper topic management without sacrificing content fluency. In practice, topic accuracy is more important than perplexity in consideration of the generated responses are already fluent with the perplexity of 36.82. Besides, comparing to DKGT *w/all Know*, our model performs better on both perplexity and BLEU-1. This is because proper topic prediction highly reduces the difficulty of picking suitable knowledge, thus facilitate response generation. Note that it is reasonable for DKGT *w/all Know* to get the highest Dist-1 score since it encounters the whole knowledge base.

4.6 Manual Evaluation

To better evaluate the generated responses, we further perform a manual evaluation. We randomly sampled 200 posts from the test frequent and rare set (50 posts for each knowledge setting) respec-

[5] https://pytorch.org/

[6] https://huggingface.co/

Opponent	Win	Loss	Tie
DKGT vs. Seq2Seq-TF***	**39.0%**	22.5%	38.5%
DKGT vs. Wizard-TF*	**36.0%**	30.0%	34.0%
DKGT vs. DKGT w/*all Know****	**35.5%**	21.0%	43.5%
DKGT vs. DKGT w/o *Graph****	**42.0%**	17.0%	41.0%

Table 5: Manual evaluation results. We conducted two-tailed binomial test to obtain the p-value. * refers to $p < 0.05$, ** refers to $p < 0.01$ and ** refers to $p < 0.001$ respectively.

tively to conduct a pair-wise comparison between DKGT and one of the other four baselines.

Annotation settings: Three annotators are asked to evaluate these 800 pairs independently with the following rules: (1) Given a post and relevant topics, annotators are required to rate among 'win', 'lose' and 'tie' (response1 versus response2) on two generated responses. (2) Model identifiers are masked during annotation. (3) If three annotators give three distinct answers, the result will be counted as 'tie'. Before the annotation starts, annotators were trained with a few examples to understand three criteria comprehensively:

- **Topic appropriateness:** whether a generated response appropriately deepens or widens the current conversation topic smoothly.
- **Content coherency:** whether a response is relevant and fluent to the given dialog history and the knowledge base.
- **Response informativeness:** whether a response is diverse and informative like produced by humans.

Annotators attained a Krippendorff's α of 0.469 on 200 mutually-labeled pairs, indicating moderate agreement.

Results: The results are shown in Table 4. It can be seen that our proposed model outperforms all the other baselines significantly. Besides, both the knowledge retriever and the dynamic graph builder boost the generated responses to become more acceptable to humans with a percentage of 14.5% and 25%, indicating that our model can better perform topic management and response generation with these proposed strategies.

It is worth noting that when comparing to DKGT *w/o Graph*, DKGT got the highest "Win" rate (42.0%) and the lowest "Lose" rate (17%), yet automatic evaluation results show that they have similar perplexity scores. By checking the annotation examples, we found that in most cases, though DKGT *w/o Graph* could predict the correct topic label for the next turn, it fails to pick a suitable knowledge candidate from the decreased knowledge base since it does not store relationships between topic entities. Moreover, DKGT *w/o Graph* tends to explore topics abruptly, while coherent transitions usually appear in DKGT's responses, which further proves the effectiveness of the dynamic graph module.

Model	Config	PPL	Acc
Wizard-TF	A	36.37	0.290
	B	37.81	0.312
	C	33.24	0.344
	D	40.63	0.295
DKGT w/o *Graph*	A	34.91	0.778
	B	36.36	0.770
	C	31.92	0.759
	D	38.47	0.758
DKGT	A	34.94	0.786
	B	36.37	0.776
	C	32.18	0.768
	D	38.60	0.781

Table 6: Automatic evaluation results under different knowledge settings. Config A and B are asymmetric settings while C and D are symmetric settings.

Opponent	Config	Win	Loss	Tie
DKGT vs. Wizard-TF	A	42%	28%	30%
	B	44%	20%	36%
	C	30%	26%	44%
	D	32%	42%	26%
DKGT vs. DKGT w/o *Graph*	A	40%	20%	40%
	B	52%	12%	36%
	C	38%	14%	48%
	D	38%	22%	40%

Table 7: Manual evaluation results under different knowledge settings.

4.7 Analysis of results under different knowledge settings

To examine our model's ability to conduct conversations under asymmetric knowledge settings, we further perform experiments under different configs between three representative models. Topical-Chat equips four types of prior knowledge settings between two conversational partners naming config A, B, C and D, where config A and config B represent asymmetric in entity-level fun facts and entity-level Wikipedia descriptions respectively. For automatic evaluation, we split the test set based on different configs and calculate corresponding perplexity and topic accuracy scores. For manual evaluation, we directly obtain the "win", "lose" and "tie" rates from our annotation results.

As shown in Table 5, DKGT and DKGT *w/o Graph* beats Wizard-TF on perplexity steadily under the four different knowledge settings, indicating that the topic predictor, as well as knowledge retriever can help with picking suitable knowledge to generate responses regardless of the symmetry of knowledge between two partners. Although DKGT has higher perplexity scores than DKGT *w/o Graph* with all the configs due to its consideration on topic accuracy, the gaps under config A and config B are much smaller, demonstrating that our system can use dynamic graphs to capture semantic information from the dialog history, then facilitates context generation. Moreover, our model still keeps the highest topic accuracy under all knowledge settings with a relatively high score on the two asymmetric datasets, which further proves that logical information stored in dynamic graphs can assist our model to manage chatting topics more appropriately.

Manual evaluation results in Table 6 also clarify the importance of the dynamic graph module when handling asymmetric knowledge bases. Comparing to DKGT *w/o Graph*, DKGT has an average "win" rate of 46% on asymmetric sets, while the value drops to 38% on the other two sets. Also, the average "loss" and "tie" rates on asymmetric sets decrease correspondingly (16% versus 18% and 38% versus 44%). These results further illustrate that our proposed dynamic graph module could facilitate the model to perform topic transition smoothly then generates more human-like responses, especially when the prior knowledge between two partners is not equal.

5 Conclusion and Future Work

In this paper, we propose a dynamic knowledge graph-based topical conversation model (DKGT) to perform coherent topic transitions under both symmetric and asymmetric knowledge settings. Specifically, a dynamic graph builder that constructs knowledge graphs from the context to form logical relationships between known and unknown topics is introduced to assist topic management. Automatic and manual evaluation results show that DKGT can not only schedule dialog topics properly but also generate informative responses preferred by humans.

In the future, we will further explore the usage of our proposed dynamic knowledge graph strategy to improve chatbot's interpretability, which may depend on some inferential methods like multi-hop reasoning. We release our codes at https://github.com/wujunjie1998/DKGT.

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.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Nouha Dziri, Ehsan Kamalloo, Kory Mathewson, and Osmar R Zaiane. 2019. Augmenting neural response generation with context-aware topical attention. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 18–31.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In *INTERSPEECH*, pages 1891–1895.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of EMNLP-IJCNLP: System Demonstrations*, pages 169–174.

Ben Hixon, Peter Clark, and Hannaneh Hajishirzi. 2015. Learning knowledge graphs for question answering through conversational dialog. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 851–861.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

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

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016b. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *Advances in neural information processing systems*, pages 9725–9735.

Zekang Li, Cheng Niu, Fandong Meng, Yang Feng, Qian Li, and Jie Zhou. 2019. Incremental transformer with deliberation decoder for document grounded conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 12–21.

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.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Yi-Lin Tuan, Yun-Nung Chen, and Hung-yi Lee. 2019. Dykgchat: Benchmarking dialogue generation grounding on dynamic knowledge graphs. *arXiv preprint arXiv:1910.00610*.

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.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Wenjie Wang, Minlie Huang, Xin-Shun Xu, Fumin Shen, and Liqiang Nie. 2018. Chat more: Deepening and widening the chatting topic via a deep model. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 255–264.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive human-machine conversation with explicit conversation goal. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3794–3804.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Jun Xu, Haifeng Wang, Zhengyu Niu, Hua Wu, and Wanxiang Che. 2020. Knowledge graph grounded goal planning for open-domain conversation generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9338–9345.

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*.

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.