Knowledge-Consistent Dialogue Generation with Knowledge Graphs

Minki Kang [*] [1] [2] **Jin Myung Kwak** [*] [3] **Jinheon Baek** [*] [1] **Sung Ju Hwang** [1] [2]

Abstract

We propose a framework for generating knowledge consistent and context-relevant dialogues with a knowledge graph (KG), named **SU**bgraph **R**etrieval-augmented **GE**neration (**SURGE**). First, our method retrieves the context-relevant subgraph from the KG, and then enforces consistency across the facts by perturbing their word embeddings conditioned on the retrieved subgraph. Then, it learns the latent representation space using graph-text multi-modal contrastive learning which ensures that the generated texts have high similarity to the retrieved subgraphs. We validate the performance of our SURGE framework on the OpendialKG dataset and show that our method generates high-quality dialogues that faithfully reflect the knowledge from the KG.

1. Introduction

Dialogue systems aim at generating human-like responses, considering the context and history of the dialogue. Recently, with the development of pre-trained language models (PLMs) for text generation (Radford et al., 2019; Raffel et al., 2020), neural dialogue agents are able to generate fluent responses. However, they often generate factually incorrect responses due to a lack of explicit knowledge. The problem can become worse, when the conversation requires accurate knowledge about certain subjects.

While retrieving the documents from a large-scale text corpus (e.g. Wikipedia) with information retrieval boosts the performance of dialogue agents (Karpukhin et al., 2020; Lewis et al., 2020b), the computational overhead of searching for the relevant documents and embedding them on the fly could be high. Thus, we instead consider the precompiled Knowledge Graph (KG) (Bollacker et al., 2008; Vrandecic & Krötzsch, 2014) consisting of symbolic facts, which represent the entities as nodes and their relations as edges, in the form of a triplet, e.g., *(Pride & Prejudice, written by, Jane Austen)*. Such KG-augmented dialogue generation models are highly efficient compared to retrieving from and augmenting with unstructured texts. This is because we can directly retrieve entities from the context without searching for all candidate documents from a large text corpus, and the retrieved facts succinctly encode the required knowledge in the most compact and effective form.

Few recent works (Tuan et al., 2019; Galetzka et al., 2021; Zhou et al., 2021) use the KG to provide facts associated with the entities in the dialogue context to the conversation agents. However, they utilize all the triplets associated to the given entity, whose facts are mostly irrelevant to the dialogue context, which could mislead the model into generating factually incorrect responses. Moreover, it is not straightforward to combine the representations from two heterogeneous modalities: the dialogue context is represented as a text, meanwhile, the knowledge is represented as a graph, i.e., handling two different modalities is non-trivial.

In this work, we tackle such challenging and fundamental issues of knowledge-consistent dialogue generation with KG[1]. In particular, we propose a context-relevant subgraph retrieval that retrieves only the relevant triplets from a large KG. Notably, our subgraph retrieval method is end-to-end trainable jointly with the generation objective (Lewis et al., 2020b). Then, to encode the retrieved subgraph along with the text sequence, we propose a graph encoding that is permutation and relation-inversion invariant yet efficient. Furthermore, to ensure that the model does make use of the encoded knowledge when generating responses, we propose a multi-modal contrastive learning objective to enforce the consistency across the retrieved facts and the generated texts. We refer to our framework as **SU**bgraph **R**etrieval-augmented **GE**neration (**SURGE**).

We validate our SURGE on the OpendialKG (Moon et al., 2019) dataset against relevant baselines, with our proposed performance metric, named Knowledge-verifying Question Answering (KQA) for accurate knowledge verification. The

[*] Equal contribution [1] Graduate School of AI, Korea Advanced Institute of Science and Technology, Daejeon, South Korea [2] AITRICS, South Korea [3] School of Computing, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. Correspondence to: Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

ICML workshop on Knowledge Retrieval and Language Models, Baltimore, Maryland, USA, 2022. Copyright 2022 by the author(s).

[1] In this work, we denote the knowledge as facts (i.e., a set of triplets) in the knowledge graph.

experimental results show that SURGE generates responses that not only agree with the gold knowledge but are also consistent with the retrieved knowledge from the KG.

Our main contributions can be summarized as follows:

- We propose a context-relevant subgraph retrieval method to extract only the relevant piece of the knowledge for the given context from the entire knowledge graph, for generating appropriate responses to the ongoing conversation.
- We propose a permutation and relation-inversion invariant yet efficient graph encoder and a multi-modal graph-text contrastive learning objective to ensure that the generated responses faithfully reflect the retrieved knowledge.
- We validate our SURGE framework against relevant baselines, demonstrating its efficacy in generating responses that are more informative by retrieving and reflecting the relevant knowledge from the KG.

2. Related Work

Knowledge-Grounded Dialogue The sources of external knowledge can be categorized into two types: documents from large unstructured corpora such as Wikipedia (Dinan et al., 2019) or Web (Nakano et al., 2021), and symbolic facts from knowledge graphs (Bollacker et al., 2008; Vrandecic & Krötzsch, 2014). Knowledge graph-augmented dialogue generation models, which use structured graphs, are more efficient than the previous methods utilizing unstructured texts (Li et al., 2020; Shuster et al., 2021), and consequently more preferable when responsiveness is important or a large unstructured text corpus is unavailable. Regarding the dialogue generation with the knowledge graph (KG), Moon et al. (2019) introduce a knowledge-grounded dialogue dataset where each dialogue comes with the large-scale KG. Tuan et al. (2019) and Zhou et al. (2021) are sequence-to-sequence models that condition the output distribution for word generation with the entities from the KG. Further, Galetzka et al. (2021) propose an efficient way to encode all of the facts in the 1-hop neighbors of the entities that appear in the dialogue history in the given KG. However, all of above methods simply match and retrieve all the facts for entities including irrelevant ones that appear in the dialogue history. Our work differs from them, since we aim at retrieving only the context-relevant subgraph among the 1-hop facts with an end-to-end trainable graph retriever.

3. Method

We first formalize the problem, and describe the key components for our **SU**bgraph **R**etrieval-augmented **GE**neration (**SURGE**) framework: context-relevant subgraph retrieval, invariant graph encoding, and graph-text contrastive learning. For preliminaries, please refer to **Appendix** C.1.

3.1. Problem Statement

Given a dialogue history $\boldsymbol{x} = [x_1, \ldots, x_N]$, a generative PLM first encodes the input tokens, and then models a probabilistic distribution $p(\boldsymbol{y}|\boldsymbol{x})$ to generate an output response $\boldsymbol{y} = [y_1, \ldots, y_T]$. This problem requires a piece of specific knowledge graph for continuing the conversation.

To this end, given a dialogue history $\boldsymbol{x}$, we first aim at retrieving a subgraph $\mathcal{Z} \subseteq \mathcal{G}$ consisting of a set of triplets $z \in \mathcal{Z}$ where $z = (\mathsf{e}_h, \mathsf{r}, \mathsf{e}_t)$, which encodes relevant knowledge of the ongoing conversation. Therefore, the distribution of the context-relevant facts $\mathcal{Z}$ is denoted as $p(\mathcal{Z}|\boldsymbol{x})$, and the likelihood of knowledge-consistent response generation then becomes $p(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z})$. To maximize this likelihood, we treat $\mathcal{Z}$ as a latent variable and then marginalize the likelihood over all possible latent variables for $\mathcal{Z}$, as follows:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{\mathcal{Z} \subseteq \mathcal{G}} p_\phi(\mathcal{Z}|\boldsymbol{x})\, p_\theta(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z}) \\ &= \sum_{\mathcal{Z} \subseteq \mathcal{G}} p_\phi(\mathcal{Z}|\boldsymbol{x}) \prod_{t}^{T} p_\theta(y_t|\boldsymbol{x}, \mathcal{Z}, \boldsymbol{y}_{1:t-1}), \end{aligned} \tag{1}$$

where, in other words, $p_\phi(\mathcal{Z}|\boldsymbol{x})$ is an output distribution of the context-relevant subgraph retriever, and $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z})$ is the target distribution of a knowledge-augmented generator, which are parameterized by ϕ and θ, respectively.

3.2. Context-Relevant Subgraph Retriever

We assume that a retrieval probability of each triplet in $\mathcal{Z} = \{z_1, \ldots, z_n\}$ is independent. Then, we decompose $p(\mathcal{Z}|\boldsymbol{x})$ into $p(z_1|\boldsymbol{x})p(z_2|\boldsymbol{x}) \ldots p(z_n|\boldsymbol{x})$.

We can now focus on retrieving the only one triplet. Therefore, we define the retrieval of one triplet with an inner product of dense vectors between the dialogue history $\boldsymbol{x}$ and the candidate triplet z as follows:

$$p_\phi(z|\boldsymbol{x}) \propto \exp(d(z)^\top q(\boldsymbol{x})), \tag{2}$$

where d is a triplet embedding function and q is a dialogue context embedding function. We use a PLM for implementing q, but we need another method for d that can reflect the property of the graph. Therefore, we propose the GNN-based triplet embedding method for realizing d. We include details on the triplet embedding d in **Appendix** C.2.

3.3. Invariant Graph Encoding

In this subsection, we specify $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z})$ which generates $\boldsymbol{y}$ conditioned on a text $\boldsymbol{x}$ and a retrieved subgraph $\mathcal{Z}$. For multi-relational graph $\mathcal{Z}$, it is important to obtain the permutation invariance and relation-inversion invariance when we encode it into the text sequence. For definitions of both invariance properties, please refer to **Appendix** C.3.

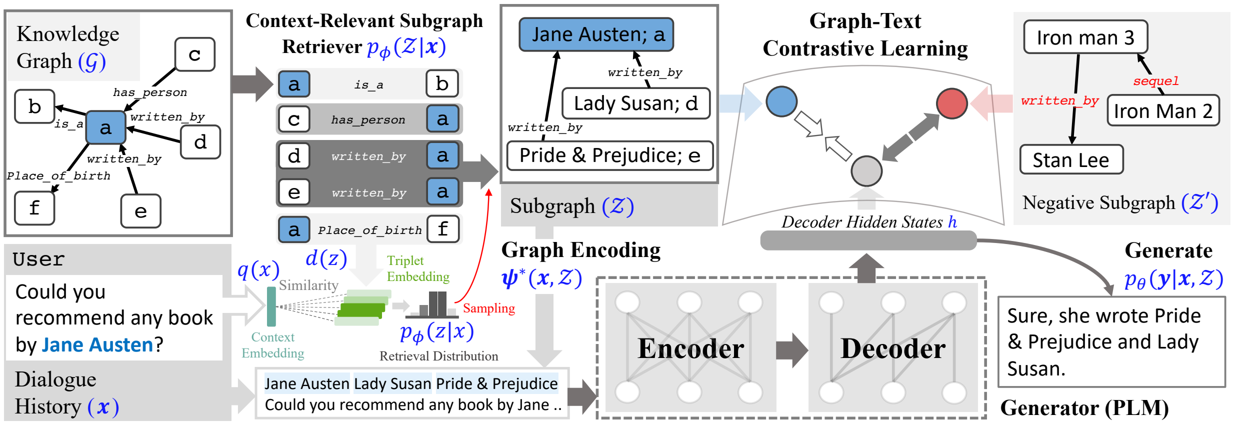

Knowledge Graph ($\mathcal{G}$)
c
has_person
b
is_a
a
written_by
d
written_by
Place_of_birth
f
e
Context-Relevant Subgraph Retriever $p_\phi(\mathcal{Z}|\boldsymbol{x})$
a is_a b
c has_person a
d written_by a
e written_by a
a Place_of_birth f
User
Could you recommend any book by Jane Austen?
Dialogue History ($\boldsymbol{x}$)
$q(x)$
$d(z)$
Similarity
Triplet Embedding
Context Embedding
Sampling
$p_\phi(z|x)$
Retrieval Distribution
Jane Austen; a
written_by
Lady Susan; d
written_by
Pride & Prejudice; e
Subgraph ($\mathcal{Z}$)
Graph Encoding
$\psi^*(\boldsymbol{x}, \mathcal{Z})$
Jane Austen Lady Susan Pride & Prejudice
Could you recommend any book by Jane ..
Graph-Text Contrastive Learning
Decoder Hidden States h
Iron man 3
sequel
written_by
Iron Man 2
Stan Lee
Negative Subgraph ($\mathcal{Z}'$)
Encoder
Decoder
Generate
$p_\theta(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z})$
Sure, she wrote Pride & Prejudice and Lady Susan.
Generator (PLM)

Figure 1: **Framework Overview.** Our framework, SURGE, consists of three parts. First, a context-relevant subgraph retriever $p_\phi(\mathcal{Z}|\boldsymbol{x})$ retrieves the subgraph $\mathcal{Z}$ relevant to the given dialogue history $\boldsymbol{x}$ from a knowledge graph $\mathcal{G}$ (e.g., 1-hop KG from entity *Jane Austen*; a). Specifically, we measure the similarity of a context and triplet embedding to compose the retrieval distribution $p_\phi(z|\boldsymbol{x})$ (§ 3.2). Then, we encode the retrieved subgraph $\mathcal{Z}$ into the input of the generator, using the graph encoding function $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z})$ (§ 3.3). Finally, we use a contrastive learning to enforce the model to generate a consistent response with the retrieved subgraph (§ 3.4).

For instance, given a sequence $\boldsymbol{x} = [x_1, \dots, x_N]$ and a subgraph $\mathcal{Z} = \{(\mathsf{a},\mathsf{d},\mathsf{b}), (\mathsf{b},\mathsf{e},\mathsf{a}), (\mathsf{a},\mathsf{d},\mathsf{c})\}$ from the retriever, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) = f([a, d, b, b, e, a, a, d, c, x_1, ..., x_N])$ with $a = q_e(\mathsf{a})$, $b = q_e(\mathsf{b})$, $c = q_e(\mathsf{c})$, $d = q_r(\mathsf{d})$, $e = q_r(\mathsf{e})$, which we term as the naïve encoding. Due to its simplicity, it is widely used for a text-conditioned generation (Lewis et al., 2020b). However, it violates two important invariance properties for graph encoding mentioned above.

To build the graph encoding method that efficiently satisfies both properties, we first only encode unique entities in front of the text sequence as follows:

$$\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))) = f([a, b, c, x_1, \dots, x_N]),$$

where ENT operator to obtain only entities from triplets, and SORT operator to sort entities in alphabetical order. However, above encoding does not consider the relational information in $\mathcal{Z}$. Therefore, we further utilize the function $\boldsymbol{\beta}$ which perturbs the entities' token embeddings with respect to their relational representations in $\mathcal{Z}$. To sum up, our invariant and efficient graph encoding is formalized as follows:

$$\boldsymbol{\psi}^*(\boldsymbol{x}, \mathcal{Z}) = \boldsymbol{\beta}(\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))), \texttt{INV}(\mathcal{Z})).$$

For more details, please refer to **Appendix** C.3 and D.

3.4. Consistent Generation with Graph-Text Contrastive Learning

We further introduce a graph-text contrastive learning method to make model generate knowledge-consistent responses. For a single pair of a graph and text,

$$\begin{aligned}\mathcal{L}_{cont} = & \frac{1}{2}\log\frac{\exp(\texttt{sim}(\zeta(\boldsymbol{z}), \xi(\boldsymbol{h}))/\tau)}{\sum_{\boldsymbol{h}'}\exp(\texttt{sim}(\zeta(\boldsymbol{z}), \xi(\boldsymbol{h}'))/\tau)} \\ & + \frac{1}{2}\log\frac{\exp(\texttt{sim}(\zeta(\boldsymbol{z}), \xi(\boldsymbol{h}))/\tau)}{\sum_{\boldsymbol{z}'}\exp(\texttt{sim}(\zeta(\boldsymbol{z}'), \xi(\boldsymbol{h}))/\tau)},\end{aligned} \quad (3)$$

where $\boldsymbol{z} = \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{z}'_i$ is the mean of graph representations, $\boldsymbol{h} = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{h}_t$ is the mean of decoder representations, sim is the cosine similarity, ζ and ξ are linear projection layers, and τ is a temperature parameter. Furthermore, $\sum_{\boldsymbol{h}'}$ and $\sum_{\boldsymbol{z}'}$ indicate the summation over negative samples in the batch (Radford et al., 2021). With Eq. 3, the model can embed the correlated pairs closer together in order to generate a consistent response to a given graph.

3.5. Training

Our whole end-to-end training objective for retrieval-augmented generation is defined as follows:

$$\mathcal{L}_{ret} = \log\sum_{i=1}^{k} p_\phi(\mathcal{Z}_i|\boldsymbol{x})p_\theta(\boldsymbol{y}|\boldsymbol{x}, \mathcal{Z}_i),\ \mathcal{Z}_i{\sim}p_\phi(\mathcal{Z}|\boldsymbol{x}), \quad (4)$$

where we simplify the sampling over n triplets as the sampling over the subgraph distribution $p_\phi(\mathcal{Z}|\boldsymbol{x})$. We assume that we can access the gold subgraph for some data in training. Thus, we further add the supervised retrieval loss to introduce a semi-supervised retriever learning as follows:

$$\mathcal{L}_{sup} = \log p_\phi(\mathcal{Z}^*|\boldsymbol{x}), \quad (5)$$

where $\mathcal{Z}^*$ is the available ground-truth subgraph. Combining all objectives in Eq. 3, 4, and 5, our final training objective is then defined as follows: $\mathcal{L} = \mathcal{L}_{ret} + \mathcal{L}_{sup} + \mathcal{L}_{cont}$.

Table 1: Experimental results on OpendialKG dataset. † indicates the model under oracle setting using the gold facts even in the test time.

		KQA		BLEU				ROUGE			Unigram
	Method	EM	F1	B-1	B-2	B-3	B-4	R-1	R-2	R-L	F1
Baselines	**No Knowledge**	7.62	13.2	15.79	9.19	5.61	3.43	19.67	7.13	19.02	22.21
	All Knowledge	30.06	34.95	15.95	9.98	6.72	4.65	20.96	8.50	20.21	24.34
	Space Efficient *(series)*	26.88	31.15	16.15	10.03	6.66	4.50	21.15	8.56	20.44	24.55
	Space Efficient *(parallel)*	28.90	33.19	16.33	10.22	6.81	4.64	21.42	8.85	20.68	24.87
	EARL	24.52	27.09	11.49	6.34	4.06	2.75	15.36	4.37	14.61	16.88
Retrieval variants	**Random Retrieval**	21.05	26.09	15.70	9.52	6.12	3.99	20.21	7.88	19.55	23.28
	Sparse Retrieval (BM25)	19.32	24.55	15.63	9.44	6.05	3.96	20.05	7.67	19.37	23.10
	Text-based Retrieval	31.00	35.95	16.87	10.64	7.23	5.07	20.63	8.53	19.89	24.16
Ours	**SURGE** *(unsupervised)*	37.35	42.24	18.10	11.65	7.99	5.59	22.14	9.50	21.23	25.91
	SURGE *(semi-supervised)*	**39.57**	**44.13**	**18.21**	**11.74**	**8.08**	**5.68**	22.11	9.41	21.22	25.91
	SURGE *(contrastive)*	39.52	43.96	17.72	11.53	7.96	5.61	**22.19**	**9.77**	**21.34**	**25.94**
Oracle	**Gold Knowledge**†	49.76	53.41	18.47	12.79	9.32	6.92	24.93	11.97	24.03	28.82
	Gold Response	83.88	86.22	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

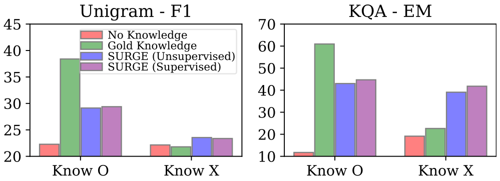


Figure 2: Results of whether gold knowledge exists (Know O) or not (Know X) for the dialogue history. We note that T5 + Gold Knowledge exactly uses the gold knowledge for generating responses – Oracle.

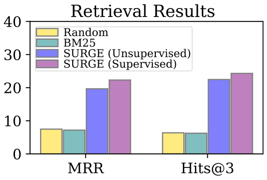


Figure 3: Knowledge retrieval results on the OpendialKG dataset, with MRR and Hits@3 as metrics.

Table 2: Results on knowledge-consistent response generation, where we compare three variants of our SURGE – unsupervised, semi-supervised and contrastive, on unigram F1 and KF1 as metrics.

Method	Unigram F1	KF1
SURGE (unsupervised)	27.78	24.09
SURGE (semi-supervised)	**28.30**	26.38
SURGE (contrastive)	28.17	**27.58**

4. Experiment

We conduct experiments on the **OpendialKG** dataset (Moon et al., 2019), which is a dialogue corpus associated with a large-scale Knowledge Graph (KG), namely Freebase (Bollacker et al., 2008). We use **T5-small** (Raffel et al., 2020) for all experiments. For details, see **Appendix** E.

4.1. Baselines

We compare different variants of our SURGE framework against various KG-augmented dialogue generation models. **No Knowledge.** This model is only provided with the dialog history, thus no external knowledge is used. **All Knowledge.** This model is provided with entire facts within a 1-hop subgraph of entities associated with the dialog history. **Gold Knowledge.** This model is provided with the exact gold knowledge, even in the test time if the gold knowledge exists. **Space Efficient Encoding.** This model takes all facts from the 1-hop subgraph of the entities as input. We use two different encoding methods introduced in (Galetzka et al., 2021), namely Space Efficient (series) and Space Efficient (parallel). **EARL.** This is an RNN-based model, where the entities are conditioned in response generation (Zhou et al., 2021). **Random/Sparse Retrieval.** These models are provided with selected facts from a 1-hop subgraph, via the random sampling or the sparse retrieval – BM25 (Robertson & Zaragoza, 2009). **Text-based Retrieval.** This model uses a pre-trained language model as the triplet embedding function of the retriever similar to (Humeau et al., 2020), instead of using GNN. **SURGE (unsupervised).** Ours with retrieved context-relevant facts from 1-hop subgraph, where the retrieval is trained without any supervision. **SURGE (semi-supervised).** Ours but the retriever is trained with supervision if the data has a gold fact. **SURGE (contrastive).** Our full model jointly trains the retriever in a semi-supervised manner with the contrastive learning term. By default, all our models are trained with an invariant and efficient graph encoding.

4.2. A Novel Metric: Knowledge-verifying QA

Existing automatic evaluation metrics, namely BLEU and ROUGE (Papineni et al., 2002; Lin, 2004), are limited in that they only consider the lexical overlaps. To solve this issue, we propose **K**nowledge-verifying **Q**uestion **A**nswering (**KQA**) which measures whether generated responses contain factually correct knowledge given the dialogue history. Compared to the existing metrics using question generation methods (Honovich et al., 2021; Wang et al., 2020), we automatically derive QA pairs for evaluation from the dialogue and the large-scale KG (Bollacker et al., 2008). For more details on KQA, please refer to **Appendix** E.1.

4.3. Experimental Results and Analyses

In Table 1, we report the knowledge-grounded response generation performances of baselines and our SURGE. Our models significantly outperform all the baselines on all metrics. The high BLEU, ROUGE, and F1 refer that ours suf-

Table 3: Performance comparisons of variants of graph encodings, described in Section 3.3.

Method	KQA		Knowledge
	EM	F1	Length
Naïve	38.18	42.18	62
Invariant	39.54	43.28	117
Efficient (entity only)	38.80	43.06	39
Invariant & Efficient	**39.57**	**44.13**	**39**

Table 4: Human evaluation on **Consis**tency, **Info**rmativeness, and **Fluency**.

Method	Consis.	Info.	Fluency
All Knowledge	**2.52**	1.99	2.62
Space Efficient	2.47	1.75	2.46
SURGE *(ours)*	**2.71**	**2.39**	**2.92**

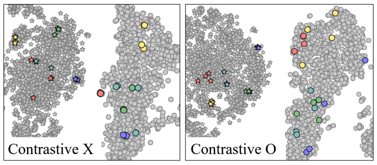


Figure 4: Visualization of the embedding space learned using our graph(star)-text(circle) contrastive learning.

ficiently learns the syntactic and semantic structure of the responses. On the other hand, high KQA scores indicate that the generated responses are formed with the correct facts, which are relevant to the dialog context. Even baselines like *All Knowledge*, *Space Efficient Encoding* (Galetzka et al., 2021), and *EARL* (Zhou et al., 2021), which are provided with all of 1-hop facts, underperform ours. The result demonstrates that retrieving relevant knowledge is critical for successful response generation. Among retrieval variants, our models achieve the best performance on all of metrics. The results indicate the use of the graph-structured information is important to retrieve the relevant facts.

In Figure 2, we further examine the generation performance by categorizing the data into two groups: ones with and without the gold knowledge. Our method notably shows notable performance even with the retrieved knowledge when there is no exact gold knowledge provided.

Knowledge Retrieval Figure 3 shows the performances of retrieval methods, where we only measure the retrieval performance on data that contain the gold knowledge. Our SURGE has a differentiable retriever, whereas *Random* and *BM25* (Robertson & Zaragoza, 2009) retrieve the fact without learning. Therefore, our models outperform both approaches by a large margin. We provide the retrieval examples for baselines and our model in Figure 9 of Appendix H.

Knowledge-Consistent Generation We conduct an ablation study on our models to validate the knowledge consistency performance of the response generation by computing the Knowledge F1 (KF1) score (Shuster et al., 2021). The KQA scores capture the overall performance of both retrieval and generation and implicitly quantify the knowledge consistency between the retrieved knowledge and generated responses. They concentrate more on evaluating the factual correctness of the generated responses. To solely focus on the response generation performance where a given knowledge is consistently reflected in the generated responses, we use the gold knowledge instead of the retrieved knowledge and randomly replace them with unseen combinations of triplets in dialogues. The newly formed knowledge ensures our models to generate genuinely from the given knowledge while not depending on the information learned from the training dataset. As shown in Table 2, our model with graph-text contrastive learning loss performs the best in the KF1 and comparable F1 to our semi-supervised model. The high KF1 score infers that the generated responses faithfully reflect the encoded knowledge.

Sensitive Analysis on Graph Encoding We further conduct an analysis on graph encoding variants introduced in Section 3.3. The knowledge length in Table 3 indicates the average token length used for graph encoding. Our *Invariant & Efficient* ψ^* performs the best against other variants, while using the lesser space at the graph encoding phase. Notably, *Invariant* achieves a comparable performance against *Invariant & Efficient*, but yields a longer sequence.

Human Evaluation We sample 30 responses of SURGE, *All Knowledge*, and *Space Efficient* on the OpendialKG test dataset (Moon et al., 2019), then conduct a human study. We recruit 46 annotators, and ask them to evaluate the quality of the generated responses by each model given in a random order, with 3 criteria – consistency, informativeness, and fluency – using a 3 point Likert-like scale. As shown in Table 4, ours obtains significantly (p-value < 0.05) higher scores than others in all criteria, which is another evidence that our framework generates consistent, informative, and fluent responses.

Embedding Space Visualization We further visualize the multi-modal graph-text latent space in Figure 4. The visualization shows that, for the same dialogue with different subgraphs, our SURGE with graph-text contrastive learning (right) generates distinct response embeddings pertraining to different subgraphs, unlike the one without graph-text contrastive learning which shows less variety over responses for the same dialogue (left).

5. Conclusion

We proposed a novel end-to-end framework for knowledge graph-augmented dialogue generation which retrieves context-relevant subgraph, encodes a subgraph with the text, and generates knowledge-consistent responses, called as **SU**bgraph **R**etrieval-augmented **GE**neration (**SURGE**). Our results demonstrate the effectiveness of our framework in both quantitative and qualitative experiments in knowledge retrieval and response generation tasks. Our work suggests a new direction to generate informative responses for knowledge graph-based dialogue task by empirically showing the importance of retrieving the more relevant subgraph knowledge rather than using all the relevant knowledge graphs when generating knowledge-grounded responses.

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Bollacker, K. D., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pp. 1247–1250, 2008.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019. URL https://doi.org/10.18653/v1/n19-1423.

Dinan, E., Roller, S., Shuster, K., Fan, A., Auli, M., and Weston, J. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Galetzka, F., Eneh, C. U., and Schlangen, D. A corpus of controlled opinionated and knowledgeable movie discussions for training neural conversation models. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S. (eds.), *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pp. 565–573. European Language Resources Association, 2020. URL https://aclanthology.org/2020.lrec-1.71/.

Galetzka, F., Rose, J., Schlangen, D., and Lehmann, J. Space efficient context encoding for non-task-oriented dialogue generation with graph attention transformer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pp. 7028–7041. Association for Computational Linguistics, 2021.

Hamilton, W. L. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, 2020.

Honovich, O., Choshen, L., Aharoni, R., Neeman, E., Szpektor, I., and Abend, O. \$q^2\$: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In Moens, M., Huang, X., Specia, L., and Yih, S. W. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pp. 7856–7870. Association for Computational Linguistics, 2021. URL https://doi.org/10.18653/v1/2021.emnlp-main.619.

Humeau, S., Shuster, K., Lachaux, M.-A., and Weston, J. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkxgnnNFvH.

Jo, J., Baek, J., Lee, S., Kim, D., Kang, M., and Hwang, S. J. Edge representation learning with hypergraphs. *CoRR*, abs/2106.15845, 2021.

Karpukhin, V., Oguz, B., Min, S., Lewis, P. S. H., Wu, L., Edunov, S., Chen, D., and Yih, W. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 6769–6781. Association for Computational Linguistics, 2020.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 7871–7880. Association for Computational Linguistics, 2020a.

Lewis, P. S. H., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020b.

Li, L., Xu, C., Wu, W., Zhao, Y., Zhao, X., and Tao, C. Zero-resource knowledge-grounded dialogue generation. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Lin, C.-Y. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Moon, S., Shah, P., Kumar, A., and Subba, R. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In Korhonen, A., Traum, D. R., and Màrquez, L. (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 845–854. Association for Computational Linguistics, 2019.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., and Schulman, J. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint*, arXiv:2112.09332, 2021.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pp. 311–318. ACL, 2002.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, 2019.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

Robertson, S. and Zaragoza, H. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009.

Scheinerman, E. and Ullman, D. *Fractional graph theory: a rational approach to the theory of graphs*. Courier Coporation, 2011.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.

Shuster, K., Poff, S., Chen, M., Kiela, D., and Weston, J. Retrieval augmentation reduces hallucination in conversation. In Moens, M., Huang, X., Specia, L., and Yih, S. W. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pp. 3784–3803. Association for Computational Linguistics, 2021.

Tuan, Y., Chen, Y., and Lee, H. Dykgchat: Benchmarking dialogue generation grounding on dynamic knowledge 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 2019, Hong Kong, China, November 3-7, 2019*, pp. 1855–1865. Association for Computational Linguistics, 2019.

Vashishth, S., Sanyal, S., Nitin, V., and Talukdar, P. P. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information*

Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pp. 5998–6008, 2017.

Vrandecic, D. and Krötzsch, M. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014.

Wang, A., Cho, K., and Lewis, M. Asking and answering questions to evaluate the factual consistency of summaries. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 5008–5020. Association for Computational Linguistics, 2020. URL https://doi.org/10.18653/v1/2020.acl-main.450.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *EMNLP 2020 - Demos, Online, November 16-20, 2020*, pp. 38–45, 2020. URL https://doi.org/10.18653/v1/2020.emnlp-demos.6.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. Deep sets. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 3391–3401, 2017.

Zhou, H., Huang, M., Liu, Y., Chen, W., and Zhu, X. EARL: informative knowledge-grounded conversation generation with entity-agnostic representation learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pp. 2383–2395. Association for Computational Linguistics, 2021.

A. Discussion

Limitation As briefly discussed in Section H, our work is limited in multiple dimensions primarily in terms of dataset, retrieval, and generation. First, the benchmark dataset is limited. Despite the fact that there are several public Knowledge Graph (KG) available (Vrandecic & Krötzsch, 2014; Bollacker et al., 2008), only one dataset (Moon et al., 2019) provides both the diverse set of dialogue and the corresponding large-scale KG. This circumstance may limit the rigorous evaluation of our framework's adaptability in various settings. Future work may study applying our approach for a wider range of dialogue datasets based on Wikipedia (Dinan et al., 2019) by leveraging existing public large-scale KG such as Wikidata (Vrandecic & Krötzsch, 2014). Second, the search space for retrieving context-relevant subgraphs can be expanded. Our SURGE framework now runs on a 1-hop KG that is rooted to entities in the given dialogue history. Finding the entity within the text, on the other hand, necessitates precise named entity extraction and entity linking. Therefore, future work may investigate extending our approach to a framework that can retrieve the context-relevant subgraph among entire KG instead of 1-hop KG. Third, there is still room for improvement in generation quality since we generate knowledge-enhanced responses with a small-scale Pre-trained Language Model (PLM) for efficiency. Such PLMs occasionally fail to generate natural sentences with a high quality (Raffel et al., 2020). Future work could aim to improve generation quality using a small-scale PLM.

Broader Impact Our proposed knowledge-grounded dialogue generation model is essential for designing user-friendly real-world AI systems. Among various types of dialogue generation models, knowledge-grounded dialogue models are trained to interact with users and convey factual information to users in natural languages. Their conversational features can be adapted to any user interfaces that connect the bilateral interaction between human and computer. We believe that the conversational interfaces can enhance the users' experiences and reduce the users' efforts in learning how to use the systems. However, knowledge-grounded dialogue models can become vulnerable to generating offensive, harmful, or misinformation responses depending on the users or data. When deploying the models in the real world, in addition to generating realistic responses, they also need to be robust to adversarial feedback from malicious users and biases inherited in pre-training or training corpus, or else they could malfunction. Along with the quantitative and qualitative evaluations on generated responses, it is worthwhile to examine robustness of the dialogue models.

B. Notations

We organize the notations we used in Table 5.

C. Method Details

In this section, we supplement details on our method in Section 3.

C.1. Preliminaries

As we use two different modalities, namely text and graph, we first define them, and then describe the neural networks to encode them. In particular, a text is defined as a sequence of tokens $\boldsymbol{x} = [x_1, ..., x_N], \forall x_i \in \mathcal{V}$, where x_i is a token and $\mathcal{V}$ is a pre-defined vocabulary formed with specific tokenization algorithms (Sennrich et al., 2016). On the other hand, a knowledge graph (KG) is a type of multi-relational graphs $\mathcal{G} = \{(\mathsf{e}_h, \mathsf{r}, \mathsf{e}_t)\} \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where e_h and e_t are head and tail entities along with their relation r, and $\mathcal{E}$ and $\mathcal{R}$ are sets of entities and relations, respectively, i.e., $\mathsf{e}_h, \mathsf{e}_t \in \mathcal{E}$ and $\mathsf{r} \in \mathcal{R}$.

To easily access different modalities in the same framework, we define the mapping function that maps entities and relations in the KG to the tokens in the text as follows: $q_e : \mathcal{E} \rightarrow \mathcal{V}^l$ and $q_r : \mathcal{R} \rightarrow \mathcal{V}^l$. In other words, any entity $\mathsf{e} \in \mathcal{E}$ and relation $\mathsf{r} \in \mathcal{R}$ can be mapped to a sequence of l tokens $\boldsymbol{x} \in \mathcal{V}^l$: $q_e(\mathsf{e}) = \boldsymbol{x}_e$ and $q_r(\mathsf{r}) = \boldsymbol{x}_r$.

Transformer A Transformer (Vaswani et al., 2017) is a neural architecture that embeds a sequence of tokens by considering their relationships. It is a basic building block of recent PLMs (Devlin et al., 2019; Radford et al., 2019). Formally, assume that we have a sequence of tokens $\boldsymbol{x} = [x_1, ..., x_N], \forall x_i \in \mathcal{V}$, then a goal of generative transformers is to generate a sequence of tokens $\boldsymbol{y}_{<t} = [y_1, ..., y_{t-1}], \forall y_i \in \mathcal{V}$, with encoder $\texttt{Enc}$, decoder $\texttt{Dec}$ and tokens' embedding function f. Thus, a hidden state at time t for generating y_t is $\boldsymbol{h}_t = \texttt{Dec}(\texttt{Enc}(\boldsymbol{X}), \boldsymbol{Y}_{<t})$, where $\boldsymbol{X} = f(\boldsymbol{x}) = [f(x_1), ..., f(x_N)]$ and $\boldsymbol{Y}_{<t} = f(\boldsymbol{y}_{<t}) = [f(y_1), ..., f(y_{t-1})]$. We note that both $\texttt{Enc}$ and $\texttt{Dec}$ functions are **permutation sensitive**.

Table 5: A list of notations that we used for defining our method.

$\mathcal{V}$	pre-defined vocabulary of tokens for pre-trained language models (text)
$\mathcal{E}$	pre-defined vocabulary of entities (symbol)
$\mathcal{R}$	pre-defined vocabulary of relations (symbol)
$\mathtt{a}, \ldots \mathtt{z}$	knowledge graph symbols written in typewrite font
$\boldsymbol{x}$	input sequence (vector)
$x_1, \ldots, x_N$	input tokens (scalar)
$\boldsymbol{y} = [y_1, \ldots, y_T]$	output sequence and tokens
$\mathcal{G}$	multi-relational graph, such as knowledge graph
$\mathcal{Z}$	retrieved subgraph: $\mathcal{Z} \subset \mathcal{G}$
z	triplet (edge): $z \in \mathcal{Z}$
q_e	mapping function of entity symbol to sequence of tokens
q_r	mapping function of relation symbol to sequence of tokens
$q(\cdot)$	text representation function for retrieval
$d(\cdot)$	triplet representation function for retrieval
$\mathtt{Enc}$	Transformer Encoder
$\mathtt{Dec}$	Transformer Decoder
f	token (word) embedding function
θ	generator parameter
ϕ	retriever parameter
$\boldsymbol{\psi}$	set encoding function
$\boldsymbol{\beta}$	perturbation function
π	set permutation
n	the number of triplets in a retrieved subgraph $\mathcal{Z}$
k	the number of samples in a marginalization term
$\boldsymbol{z}$	encoder hidden state (single token)
$\boldsymbol{Z}$	encoder hidden states (sequence of tokens)
$\boldsymbol{h}$	decoder hidden state (single token)
$\boldsymbol{H}$	decoder hidden states (sequence of tokens)
$\boldsymbol{X}$	input embeddings after token embedding function (sequence)
$\boldsymbol{Y}$	output embeddings after token embedding function (sequence)

Graph Neural Network A Graph Neural Network (GNN) represents a node with its neighboring nodes over the graph structure (Hamilton, 2020), which is formalized as follows:

$$\boldsymbol{e}_t^{(k+1)} = \texttt{GNN}^{(k)}(\boldsymbol{e}_t^{(k)};\mathcal{G}) = \texttt{UPD}^{(k)}(\boldsymbol{e}_t^{(k)}, \texttt{AGG}^{(k)}(\{\boldsymbol{e}_h^{(k)} \mid \forall \mathsf{e}_h \in \mathcal{N}(\mathsf{e}_t;\mathcal{G})\})), \tag{6}$$

where $\boldsymbol{e}_t$ and $\boldsymbol{e}_h$ are embeddings of entities (nodes) e_t and e_h, respectively, $\mathcal{N}(\mathsf{e}_t;\mathcal{G}) = \{\mathsf{e}_h \mid (\mathsf{e}_h, \mathsf{r}, \mathsf{e}_t) \in \mathcal{G}\}$ is a set of neighboring entities of e_t, $\texttt{AGG}$ is a function that aggregates embeddings of e_t's neighboring entities, and $\texttt{UPD}$ is a function that updates a representation of $\boldsymbol{e}_t$ with the aggregated messages from $\texttt{AGG}$, at each iteration k.

C.2. Context-Relevant Subgraph Retriever

We briefly introduced the outline of our context-relevant subgraph retriever in Section 3.2. In this section, we supplement the details of the context-relevant subgraph retriever majorly on the details of the triplet embedding function d.

Let consider a set of triplets associated to the entities that appear in the given dialogue context $\{(\mathsf{e}, \mathsf{r}, \mathsf{e}_t) \text{ or } (\mathsf{e}_h, \mathsf{r}, \mathsf{e}) \mid q_e(\mathsf{e}) \subseteq \boldsymbol{x}\}$, as the retrieval candidates. To effectively represent the triplets consisting of entities and their relations as items, we use GNNs described in Section C.1 for the triplet embedding function d. In our triplet retrieval, representing both nodes and edges, which are equally essential components for the multi-relational graph, is worthwhile to represent an entire triplet. To do so, we adopt the existing edge message passing framework (Jo et al., 2021) that transforms edges of the original graph to nodes of the dual hypergraph (Scheinerman & Ullman, 2011) (i.e., transforming $\mathcal{G}$ to $\mathcal{G}^*$), which allows us to use existing node-level GNNs for representing edges of the original graph (See Section E.1 for more implementation details). Formally, our triplet embedding function is denoted as follows:

$$d(z) = \texttt{MLP}([\boldsymbol{e}_h \parallel \boldsymbol{r} \parallel \boldsymbol{e}_t]),\ \boldsymbol{e}_h = \texttt{GNN}(\boldsymbol{e}_h;\mathcal{G}),\ \boldsymbol{r} = \texttt{GNN}(\boldsymbol{r};\mathcal{G}^*),\ \boldsymbol{e}_t = \texttt{GNN}(\boldsymbol{e}_h;\mathcal{G}), \tag{7}$$

where $z = (e_h, r, e_t)$, and $\parallel$ is the concatenation operator.

C.3. Invariant Graph Encoding

In this subsection, we illustrate more details on the invariant graph encoding described in Section 3.3, including the formal definition of the graph encoding, permutation invariance, and the relation inversion invariance.

We first define the notion of graph encoding, whose goal is to leverage the retrieved subgraph information along with the dialogue history for response generation, which is formalized in Definition C.1.

Definition C.1. (Graph Encoding) *Let $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z})$ be a graph encoding function. Then, given a sequence of tokens $\boldsymbol{x} = [x_1, ..., x_N]$ and a subgraph $\mathcal{Z}$, it first yields a new sequence $\boldsymbol{x}' = [x'_1, ..., x'_m, x_1, ..., x_N]$ where $[x'_1, ..., x'_m]$ comes from $q_e(\mathsf{e}) = x'_e$ and $q_r(\mathsf{r}) = x'_r$ $\forall\ (\mathsf{e}, \mathsf{r}, *) \in \mathcal{Z}$. Then, it embeds a sequence $\boldsymbol{X}' = [f(x'_1), ..., f(x'_m), f(x_1), ..., f(x_N)] = f([x'_1, ..., x'_m, x_1, ..., x_N])$, where f is the token embedding function. Consequently, $\boldsymbol{X}' = \boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z})$.*

For instance, given a sequence $\boldsymbol{x} = [x_1, \ldots, x_N]$ and a subgraph $\mathcal{Z} = \{(\mathsf{a}, \mathsf{d}, \mathsf{b}), (\mathsf{b}, \mathsf{e}, \mathsf{a}), (\mathsf{a}, \mathsf{d}, \mathsf{c})\}$ from the retriever, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) = f([a, d, b, b, e, a, a, d, c, x_1, ..., x_N])$ with $a = q_e(\mathsf{a})$, $b = q_e(\mathsf{b})$, $c = q_e(\mathsf{c})$, $d = q_r(\mathsf{d})$, $e = q_r(\mathsf{e})$, which we term as the naïve encoding. Due to its simplicity, it is widely used for a text-conditioned generation (Lewis et al., 2020b). However, for graph encoding, it violates two important invariance properties: permutation invariance (Zaheer et al., 2017) and relation-inversion invariance, which are formalized in Definition C.2, C.3.

Definition C.2. (Permutation Invariance) *For any set permutation π, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) = \boldsymbol{\psi}(\boldsymbol{x}, \pi \cdot \mathcal{Z})$, i.e., an order of elements in a subgraph does not affect a representation.*

Definition C.3. (Relation Inversion Invariance) *Let a relation $\neg\mathsf{d}$ be an inverse relation to d, if $(\mathsf{a}, \mathsf{d}, \mathsf{b}) = (\mathsf{b}, \neg\mathsf{d}, \mathsf{a})$ $\forall \mathsf{a}, \mathsf{b} \in \mathcal{E}$. Then, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z} \cup \{(\mathsf{a}, \mathsf{d}, \mathsf{b})\}) = \boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z} \cup \{(\mathsf{b}, \neg\mathsf{d}, \mathsf{a})\})$ for any subgraph $\mathcal{Z}$.*

Invariant Graph Encoding To meet both properties, we consider two additional operations on a set of triplets up to the naïve encoding. We first define a $\texttt{SORT}$ operator that returns the same output regardless of the order of input set elements, as follows:

$$\texttt{SORT}(\pi \cdot \mathcal{Z}) = \texttt{SORT}(\pi' \cdot \mathcal{Z}),\ \forall \pi, \pi' \in S_n, \tag{8}$$

where S_n is a set of all possible permutations for n elements. Moreover, we define a $\texttt{INV}$ operator that adds the inverse triplet of each triplet in the subgraph $\mathcal{Z}$, as follows:

$$\texttt{INV}(\mathcal{Z}) = \mathcal{Z} \cup \{(\mathsf{e}_t, \neg\mathsf{r}, \mathsf{e}_h) \mid (\mathsf{e}_h, \mathsf{r}, \mathsf{e}_t) \in \mathcal{Z}\}. \tag{9}$$

With above operations, we now define a more solid graph encoding function: $\boldsymbol{\psi}(\boldsymbol{x}, \texttt{SORT}(\texttt{INV}(\mathcal{Z})))$, which satisfies both permutation and relation inversion invariance.

However, above encoding is not efficient since it requires the $\mathcal{O}(n)$ space complexity for encoding a graph with n triplets. To be more efficient, we newly define $\tilde{\boldsymbol{\psi}}$ that only encodes the unique nodes (entities) along the sequence, formalized as follows:

$$\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))) = f([a, b, c, x_1, \dots, x_N]),$$

where $\texttt{ENT}(\mathcal{Z})$ returns the set of unique nodes in $\mathcal{Z}$ and $\texttt{SORT}$ is used to preserve the permutation invariance. This encoding is thus invariant but efficient since it only costs $\mathcal{O}(k)$, for a k-entity sequence where $k < n$. However, as it does not consider the relational information in $\mathcal{Z}$, we further perturb the entities' token embeddings with respect to their representations in $\mathcal{Z}$. Specifically, for each entity $\mathsf{a} \in \texttt{ENT}(\mathcal{Z})$, we apply affine transformations from learnable Multi-Layer Perceptrons ($\texttt{MLP}$) on the token embedding of a as follows:

$$\begin{gathered}\boldsymbol{\beta}(f(a), \mathcal{Z}) = (1 + \boldsymbol{\gamma}) * f(a) + \boldsymbol{\delta}, \\ \boldsymbol{\gamma} = \texttt{MLP}_1(\boldsymbol{\eta}), \quad \boldsymbol{\delta} = \texttt{MLP}_2(\boldsymbol{\eta}), \quad \boldsymbol{\eta} = \texttt{UPD}(f(a), \texttt{AGGR}(\{f(b), \mathsf{r} \mid \forall \mathsf{b} \in \mathcal{N}(\mathsf{a}; \mathcal{Z})\})),\end{gathered} \tag{10}$$

where $\boldsymbol{\beta} : \mathbb{R}^d \to \mathbb{R}^d$ perturbs the embedding according to $\mathcal{Z}$, $\texttt{AGGR}$ is the relation-aware aggregation function for triplet $(\mathsf{b}, \mathsf{r}, \mathsf{a}) \in \mathcal{Z}$ with $a = q_e(\mathsf{a})$ and $b = q_e(\mathsf{b})$. In sum, we denote a relation-aware invariant and efficient encoder $\boldsymbol{\psi}^*$, formally represented as follows:

$$\boldsymbol{\psi}^*(\boldsymbol{x}, \mathcal{Z}) = \boldsymbol{\beta}(\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))), \texttt{INV}(\mathcal{Z})),$$

where $\boldsymbol{\beta}$ can be applied to the sequence of representations, $\boldsymbol{\beta} : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$. We conclude that our graph encoding satisfies both properties. For proofs, please see Section D.

D. Proofs

In this section, we first show that a naïve encoding function $\boldsymbol{\psi}$ in Section C.3 is neither permutation invariant nor relation inversion invariant, formalized in Proposition D.1. After that, we prove that our invariant and efficient encoding function $\boldsymbol{\psi}^*$ with graph-conditioned token embedding perturbation $\boldsymbol{\beta}$ is both permutation invariant and relation inversion invariant, formalized in Proposition D.2.

Proposition D.1. *A naïve encoding function $\boldsymbol{\psi}$ is neither permutation invariant nor relation inversion invariant.*

Proof. We prove this by contradiction.

Suppose $\boldsymbol{x} = [x_1, \dots, x_n]$ and $\mathcal{Z} = \{(\mathsf{a}, \mathsf{d}, \mathsf{b}), (\mathsf{b}, \mathsf{e}, \mathsf{a}), (\mathsf{a}, \mathsf{d}, \mathsf{c})\}$. Moreover, let $\mathcal{Z}' = \{(\mathsf{b}, \mathsf{e}, \mathsf{a}), (\mathsf{a}, \mathsf{d}, \mathsf{b}), (\mathsf{a}, \mathsf{d}, \mathsf{c})\}$ be one of permutations of $\mathcal{Z}$ with the permutation order $\pi = (2, 1, 3)$.

From the definition of naïve encoding, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) = [\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{b}, \boldsymbol{b}, \boldsymbol{e}, \boldsymbol{a}, \boldsymbol{a}, \boldsymbol{d}, \boldsymbol{c}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_n]$ and $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}') = [\boldsymbol{b}, \boldsymbol{e}, \boldsymbol{a}, \boldsymbol{a}, \boldsymbol{d}, \boldsymbol{b}, \boldsymbol{a}, \boldsymbol{d}, \boldsymbol{c}, \boldsymbol{x}_1, ..., \boldsymbol{x}_n]$. Therefore, it is easy to notice that $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) \neq \boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}')$, thus the naïve encoding is not permutation invariant.

We then show naïve encoding is not relation inversion invariant. Suppose $\mathcal{Z}'' = \{(\mathsf{a}, \mathsf{d}, \mathsf{b}), (\mathsf{b}, \mathsf{e}, \mathsf{a}), (\mathsf{c}, \neg\mathsf{d}, \mathsf{a})\}$, where $(\mathsf{a}, \mathsf{d}, \mathsf{c}) \in \mathcal{Z}$ is changed to its inverse relation $(\mathsf{c}, \neg\mathsf{d}, \mathsf{a})$. Then, $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}'') = [\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{b}, \boldsymbol{b}, \boldsymbol{e}, \boldsymbol{a}, \boldsymbol{c}, \neg\boldsymbol{d}, \boldsymbol{a}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_n]$ that is different against $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z})$: $\boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}) \neq \boldsymbol{\psi}(\boldsymbol{x}, \mathcal{Z}'')$. Therefore, the naïve encoding function is not relation inversion invariant.

In conclusion, from the above two counterexamples, we prove that a naïve encoding function $\boldsymbol{\psi}$ is neither permutation invariant nor relation inversion invariant. □

We now provide proof of the permutation invariance and the relation inversion invariance of our invariant and effective graph encoding $\boldsymbol{\psi}^*$, described in Section 3.4. Before starting the proof, we first revisit the permutation invariant property of graph neural networks that sum, mean and max operators are permutation invariant for the input set of $\texttt{AGGR}$. Thus, if we use sum, mean, or max for $\texttt{AGGR}$, then the token embedding perturbation function $\boldsymbol{\beta}$ naturally satisfies the permutation invariance property. In other words, $\boldsymbol{\beta}(\boldsymbol{X}, \mathcal{Z}) = \boldsymbol{\beta}(\boldsymbol{X}, \pi \cdot \mathcal{Z})$, where $\boldsymbol{X} = \tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z})))$ for any permutation π.

Proposition D.2. *Invariant and efficient encoding $\boldsymbol{\psi}^*$ is both permutation invariant and relation inversion invariant.*

Proof. Suppose $\boldsymbol{x} = [x_1, \ldots, x_n]$ and $\mathcal{Z} = \{(\mathsf{a}, \mathsf{d}, \mathsf{b}), (\mathsf{b}, \mathsf{e}, \mathsf{a}), (\mathsf{a}, \mathsf{d}, \mathsf{c})\}$. We first consider the permutation invariance for any permuted set $\mathcal{Z}' = \pi \cdot \mathcal{Z}$. While $\mathcal{Z}$ and $\mathcal{Z}'$ can have different orders of elements thus the outputs of $\texttt{ENT}(\mathcal{Z})$ and $\texttt{ENT}(\mathcal{Z}')$ could be different, we always obtain the same output with the usage of the $\texttt{SORT}$ operator for encoding. In other words, $\texttt{SORT}(\texttt{ENT}(\mathcal{Z})) = \texttt{SORT}(\texttt{ENT}(\mathcal{Z}'))$ holds due to the definition of the $\texttt{SORT}$ operation in Eq. 5 of the main paper. Therefore, $\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))) = \tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}')))$ holds.

Further, since the token embedding perturbation function $\boldsymbol{\beta}(\cdot, \mathcal{Z})$ along with sum, max, or mean in $\texttt{AGGR}$ is also permutation invariant with regards to any permutation on $\mathcal{Z}$, we conclude our invariant and efficient encoding $\boldsymbol{\psi}^*$ is permutation invariant.

We finally prove the relation inversion invariance property of $\boldsymbol{\psi}^*$. Suppose $\mathcal{Z}'' = (\mathcal{Z} \cup t') \setminus t$ where $t \in \mathcal{Z}$ is any triplet in a set and t' is inverse of t. Then, $\texttt{ENT}(\mathcal{Z}) = \texttt{ENT}(\mathcal{Z}'')$ that is trivial as $\texttt{ENT}(\mathcal{Z})$ returns the set of only unique nodes in $\mathcal{Z}$. Therefore, $\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))) = \tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}'')))$ correspondingly holds.

The remaining step to conclude the proof is to show the following equality: $\boldsymbol{\beta}(\cdot, \texttt{INV}(\mathcal{Z})) = \boldsymbol{\beta}(\cdot, \texttt{INV}(\mathcal{Z}''))$, to conclude that $\boldsymbol{\psi}^*(\boldsymbol{x}, \mathcal{Z}) = \boldsymbol{\psi}^*(\boldsymbol{x}, \mathcal{Z}'')$ from $\boldsymbol{\beta}(\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}))), \texttt{INV}(\mathcal{Z})) = \boldsymbol{\beta}(\tilde{\boldsymbol{\psi}}(\boldsymbol{x}, \texttt{SORT}(\texttt{ENT}(\mathcal{Z}''))), \texttt{INV}(\mathcal{Z}''))$. We note that $\texttt{INV}(\mathcal{Z}) = \texttt{INV}(\mathcal{Z}'')$, as $\texttt{INV}$ makes any graph as bidirectional one by the definition in Eq. 6 of the main paper. Therefore, $\boldsymbol{\beta}(\cdot, \texttt{INV}(\mathcal{Z})) = \boldsymbol{\beta}(\cdot, \texttt{INV}(\mathcal{Z}''))$ holds, and the relation inversion invariance property of $\boldsymbol{\psi}^*$ holds.

□

E. Experimental Setup

In this section, we introduce the detailed experimental setups for our models and baselines. Specifically, we describe the details on implementation, dataset, training and model in the following subsections of E.1, E.2, E.3 and E.4, one by one.

E.1. Implementation Details

We use the T5-small (Raffel et al., 2020) as the base Pre-trained Language Model (PLM) for all experiments. For the pre-trained checkpoint, we use the version that the authors released. For all implementations, we use Pytorch (Paszke et al., 2019). To easily implement the language model, we use the huggingface transformers library (Wolf et al., 2020).

Retriever Details In this paragraph, we describe the implementation details of our context-relevant subgraph retriever, including the triplet embedding and dialogue context embedding for the retriever.

For the dialogue history embedding function q, we use the existing pre-trained language model (PLM). Specifically, we use the encoder part of the T5-small model (Raffel et al., 2020) and freeze the parameters of it not to be trained. We then instead add a Multi-Layer Perceptron (MLP) on top of it, to give a point-wise attention (Bahdanau et al., 2015) to each token, whereby all tokens are not equally considered in the sentence encoding. Formally,

$$q(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i * \boldsymbol{z}_i, \qquad \boldsymbol{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n] = \texttt{Enc}(\boldsymbol{X}), \qquad \alpha_i = \frac{\exp(\texttt{MLP}(\boldsymbol{z}_i))}{\sum_{j=1}^{n} \exp(\texttt{MLP}(\boldsymbol{z}_j))} \; \forall i$$

where α_i is a scalar, and $\texttt{MLP}$ is a Multi-Layer Perceptron consisting of two linear layers and ReLU nonlinearity.

For obtaining triplet representations, we need to embed the entity (node) and relation (edge) into the latent space. Similar to the token embedding matrix used in PLMs, we can introduce the entity and relation embedding matrices. However, since the number of entities used in Freebase of OpendialKG (Moon et al., 2019) is too large compared to the number of tokens in T5 (100,814 vs 32,000) (Raffel et al., 2020), it is inefficient to introduce the trainable entity embedding matrix for the retriever.

Thus, we instead reuse the contextualized representation from the PLM encoder, to embed each node if the corresponding entity exists in the dialogue context. Formally, suppose that there is a triplet $\{(\mathsf{e}_h, \mathsf{r}, \mathsf{e}_t)\}$ in the 1-hop subgraph $\mathcal{G}$, which satisfies the following condition: $q_e(\mathsf{e}_h) \subseteq \boldsymbol{x}$ or $q_e(\mathsf{e}_t) \subseteq \boldsymbol{x}$. If so, we can know the position of the mapped entity within the dialogue history: $[x_{start}, ..., x_{end}] = q_e(\mathsf{e}_h)$ from $q_e(\mathsf{e}_h) \subseteq \boldsymbol{x}$. Therefore, the node embedding for the entity e_h is obtained by $\texttt{EntEmb}(\mathsf{e}_h) = \frac{1}{|q_e(\mathsf{e}_h)|} \sum_{i=start}^{end} \texttt{Enc}(\boldsymbol{X})_i$ iff $q_e(\mathsf{e}_h) \subseteq \boldsymbol{x}$. For edge embedding, we use the trainable relation embedding matrix $\boldsymbol{R} \in \mathbb{R}^{|\mathcal{R}| \times 128}$ to represent the edge, since the number of relations is relatively small (1,357).

With our node and edge representations, we now focus on representing the triplet in Eq. 4 of the main paper for its retrieval. In particular, we use the Graph Neural Networks (GNNs) for encoding triplets, where we obtain the node representations

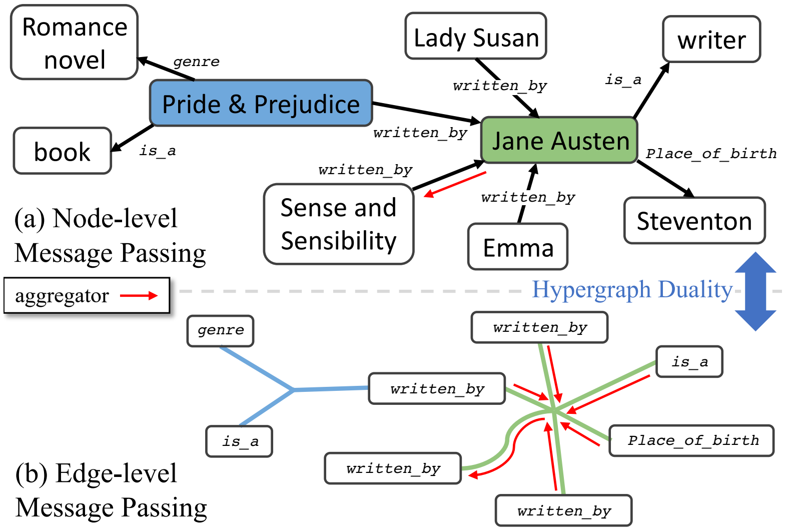


Figure 5: **Triplet Representation for Retrieval.** To represent each triplet with regards to its graph structure, we use the message passing on both nodes and edges. (a) Node-level Message Passing. To represent the entity *Sense and Sensibility*, the message from its neighbors – the entity *Jane Austen* – is aggregated. (b) Edge-level Message Passing. To represent the relation `written_by`, the messages from relations associated to a green hyperedge are aggregated. We do not draw self-loops and inverse edges for simplicity.

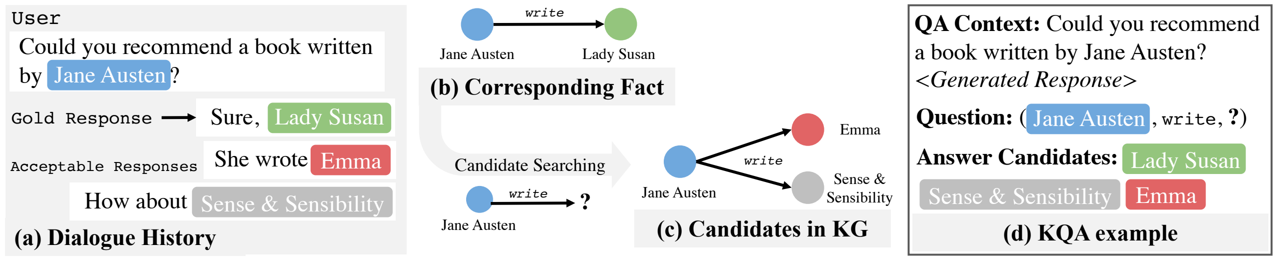


Figure 6: **KQA Diagram.** (Left) An example where multiple responses are acceptable but the gold response cannot reflect all of them. (Middle) We first find the fact from the KG that reflects the relation between entities within the user input and gold response (b), and then search candidate facts from the KG (c). (Right) Corresponding KQA example. If a generated response contains the one of answer candidates, the KQA can predict it (success).

from the Graph Convolutional Network (GCN) (Kipf & Welling, 2017) that is a widely used architecture for representing the nodes with respect to their graph structures. However, for representing the edges, we use the Edge Hypergraph Graph Neural Network (EHGNN) used in Jo et al. (2021), due to its simplicity but effectiveness for edge representations. We summarize our triplet representation in Figure 5.

Graph Encoder Details In this paragraph, we describe the implementation details of the token embedding perturbation function β used in our *Invariant and Efficient* graph encoding introduced in Section 3.4. To be aware of the relation of the graph over GNNs, we use the simplified version of CompGCN (Vashishth et al., 2020). For architectural details, instead of using the different linear layers to distinguish the inverse relation from its opposite relation, we use the same linear layer. Also, we use subtraction as the specific composition operator for reflecting relations in CompGCN.

Then, we form the learnable affine transformation based on the aggregated representation from GNN layers, to perturb the token embeddings with respect to their graph information as in Eq. 7 of the main paper. In particular,

$$\eta = \mathsf{UPD}(f(a), \mathsf{AGGR}(\{f(b), \mathsf{r} \mid \forall \mathsf{b} \in \mathcal{N}(\mathsf{a}; \mathcal{Z})\})), \quad \gamma = \mathsf{MLP}_1(\eta), \quad \boldsymbol{\delta} = \mathsf{MLP}_2(\eta),$$
$$\boldsymbol{\beta}(f(a), \mathcal{Z}) = (\mathbf{1} + \gamma) * f(a) + \boldsymbol{\delta},$$

where MLP_1 and MLP_2 are learnable MLPs consisting of two linear layers with ReLU nonlinearity.

KQA Details In this paragraph, we describe the implementation details for our Knowledge-verifying Question Answering (KQA). For building the QA dataset, we first gather the dialogue sessions where the gold response contains the entity from the whole OpendialKG dataset (Figure 6 (a)). Then, we extract the triplet from the given whole KG where the head entity is placed within the dialogue history and the tail entity is placed within the gold response (Figure 6 (b)). We build a QA training dataset based on the extracted triplets and a corresponding dialogue session. To diversify the training data, we replace the tail entity of each triplet with plausible candidate entities within KG and change the entity in the response following the changed entity on the triplet (Figure 6 (c,d)). As a result, we obtain the QA dataset size of 200k. We train the BERT-base (Devlin et al., 2019) with the constructed QA dataset. We hold out 10% of data for validation and obtain the fine-tuned BERT model with 88.89 F1 score on the hold-out validation set. When we apply the fine-tuned QA model on the evaluation of the generated responses, we rebuild the QA evaluation set with the generated response instead of a gold response as illustrated in Figure 3 of the main paper.

E.2. Dataset Details

We mainly conduct experiments on **OpendialKG** (Moon et al., 2019), which provides the parallel dialogue corpus corresponding to the existing large-scale Knowledge Graph (KG) named Freebase (Bollacker et al., 2008). The provided large-scale KG consists of total 1,190,658 fact triplets over 100,813 entities and 1,358 relations. This dataset is collected from 15K human-to-human role-playing dialogues, having multi-turns, from which we pre-process that each assistance response is the label and its corresponding dialogue history is the input. Although some of the data contain the gold knowledge that is useful for generating the response on the ongoing conversation, we found that 51% of data has no gold knowledge. To overcome this limitation, we additionally find entities from the dialogue history using the Named Entity Recognition module in spaCy[2], and then include the extracted entities' corresponding triplets in the KG to the dataset. Since the dataset does not provide the pre-defined data split, we randomly split sessions into train (70%), validation (15%), and test sets (15%).

E.3. Training Details

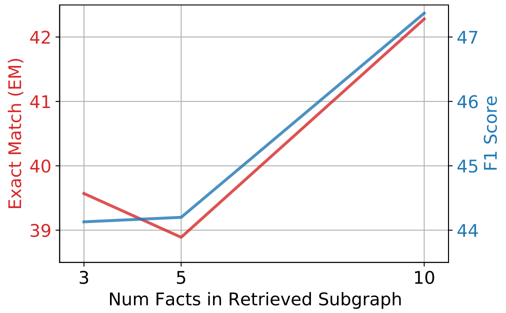
All experiments are constrained to be done with a single 48GB Quadro 8000 GPU. SURGE training needs 12 GPU hours. For all experiments, we select the best checkpoint on the validation set. We fine-tune the SURGE for 10 epochs on the training set, where we set the learning rate as 1e-4, weight decay as 0.01, learning rate decay warmup rate as 0.06, maximum sequence length for dialogue history as 256, maximum sequence length for knowledge as 128, and batch size as 24. For retrieval, we use the subgraph size n as 3, and sample size k for marginalization as 4. We use the AdamW (Loshchilov & Hutter, 2019) optimizer for training. For fairness, we apply the same training setting to all baselines if applicable.

E.4. Model Details

In this subsection, we describe the details of baselines and our models used in our experiments, as follows:

1. **No Knowledge**: This model is provided with only the dialog history. No knowledge is used to generate responses.
2. **Gold Knowledge**: This model is provided with the dialogue history along with its exact gold knowledge for the gold response. Thus, since this model uses such gold knowledge, we expect the results of it as the upper bound of the task.
3. **Space Efficient (series)**: This model is provided with all the knowledge which are related to the entities that appeared in the dialogue history (Galetzka et al., 2021), by matching the entities in the dialogue history and the entities in the KG. In particular, this model encodes the entities and their relations explicitly in the words in the encoder part.
4. **Space Efficient (parallel)**: This model is mostly the same as the above model – space Efficient (series) – except the knowledge encoding part. Specifically, it encodes the entities in the words like the above, whereas, encoding the relation between entities in the segmentation block of the entities (Galetzka et al., 2021).
5. **EARL**: This model uses the RNN-based encoder-decoder architecture with the entity-agnostic representation learning (Zhou et al., 2021), with all the provided knowledge associated with the entities in the dialogue history. Specifically, this model first calculates the probability of words obtained by encoding the entities in the KG, and then uses such probabilities to generate a word in the decoding phase.
6. **Random Retrieval**: This model is provided with entire facts from 1-hop subgraphs of entities that appeared in the dialogue history. However, instead of encoding all the knowledge in one-hop subgraph as in Space Efficient, this model randomly samples them, which are then used for generating responses.
7. **Sparse Retrieval** (BM25): This model is also provided with entire facts from 1-hop subgraphs of entities. To sample

[2] https://spacy.io/



Method	MRR	Hits@1	Hits@3	Hits@5	Hits@10	Hits@100
Random Retrieval	7.47	2.31	6.36	9.72	17.01	61.91
Sparse Retrieval (BM25)	7.17	2.22	6.23	8.98	16.36	56.88
SURGE (unsupervised)	19.66	9.55	22.46	29.81	41.09	69.35
SURGE (semi-supervised)	22.30	13.28	24.31	29.60	42.72	64.44

Figure 7: (Left:) Performances of our SURGE by varying the number of facts for retrieving the subgraph (i.e., varying the number of triplets in the subgraph) from three, to five, to ten, with EM and F1 scores of KQA as evaluation metrics. (Right:) We additionally report the knowledge retrieval performances, with MRR and Hits@K as evaluation metrics.

Table 6: Experimental results on OpendialKG dataset with **BART-base** as the base PLM.

	KQA		BLEU				ROUGE			Unigram
Method	EM	F1	B-1	B-2	B-3	B-4	R-1	R-2	R-L	F1
No Knowledge *(BART-base)*	22.87	27.53	17.38	10.79	7.16	4.81	20.64	8.22	19.92	24.36
Space Efficient *(BART-base, Series)*	38.00	42.41	18.56	11.85	8.01	5.56	22.36	9.43	21.48	26.38
Space Efficient *(BART-base, Parallel)*	39.77	43.90	18.90	12.19	8.35	5.81	**22.63**	**9.79**	**21.76**	**26.79**
SURGE *(BART-base, semi-supervised, $n = 10$)*	41.85	45.75	**19.13**	**12.37**	**8.55**	**6.09**	21.81	9.26	20.97	26.41
SURGE *(T5-small, semi-supervised, $n = 3$)*	39.57	44.13	18.21	11.74	8.08	5.68	22.11	9.41	21.22	25.91
SURGE *(T5-small, semi-supervised, $n = 10$)*	**42.28**	**47.37**	18.04	11.70	8.11	5.75	22.08	9.49	21.13	26.02

relevant facts to the dialogue history among the entire facts, this model uses BM25 (Robertson & Zaragoza, 2009) that is a sparse retrieval model. To be specific, let assume we have a dialogue history and its corresponding facts from 1-hop subgraphs of matched entities. Then, to run the BM25 algorithm, we first concatenate components of each fact consisting of two entities and one relation, and tokenize the dialogue history and the facts for obtaining corpus and queries, respectively, for BM25. After that, BM25 calculates the lexical overlapping score between the dialogue context (corpus) and the one-hop fact (query), from which we use the relevant facts having top-k scores by BM25.

8. **SURGE (unsupervised)**: Our basic subgraph retrieval-augmented generation framework that is provided with entire facts from 1-hop subgraphs of entities. In particular, this model trains the structure-aware subgraph retriever without any guidance of the gold knowledge (i.e., ground truth knowledge for the dialogue history is not given). In other words, for the given dialogue context, this model implicitly learns to retrieve the context-relevant knowledge, and then generates the response with the retrieved knowledge.
9. **SURGE (semi-supervised)**: Our subgraph retrieval-augmented generation framework with semi-supervised learning of graph retrieval, with provided entire facts from 1-hop subgraphs of entities. Unlike the unsupervised version of SURGE, this model trains the retriever to select the gold knowledge if the dialogue context has such knowledge during training.
10. **SURGE (contrastive)**: Our full subgraph retrieval-augmented generation framework with the contrastive learning of graph-text modalities as well as the semi-supervised learning of graph retrieval, with provided entire facts from 1-hop subgraphs of entities. Unlike aforementioned frameworks of ours, this additionally enforces the model to faithfully reflect the retrieved knowledge in the input, to the generated response with contrastive learning.

F. Additional Experiments

F.1. Varying the Number of Facts in Subgraphs

We experiment our SURGE framework with varying the number of facts in retrieval, which are then used in our graph encoding function to condition the encoded graph information for response generation. Specifically, in Figure 7, we report the EM and F1 scores measured by our KQA for our SURGE framework, with different numbers of facts within a retrieved subgraph: $n = [3, 5, 10]$. Note that, in this experiment, we only use the semi-supervised model without the contrastive loss. We expect that the performance of our SURGE will increase as we increase the number of facts within the retrieved subgraph,

Table 7: Experimental results on KOMODIS dataset with T5-small as the base PLM.

	BLEU				ROUGE			
	B-1	B-2	B-3	B-4	R-1	R-2	R-L	F1
No Knowledge	8.02	4.12	2.44	1.53	16.07	3.62	15.72	16.60
Random	9.45	5.30	3.48	2.47	17.60	4.50	17.20	18.57
Space Efficient *(Series)*	7.08	3.96	2.64	1.93	15.69	3.68	15.36	16.61
Space Efficient *(Parallel)*	7.71	4.45	3.00	2.20	16.61	4.16	16.27	17.65
SURGE (Ours)	**10.16**	**5.89**	**3.94**	**2.84**	**17.74**	**4.85**	**17.32**	**19.22**

since the model can leverage more numbers of knowledge for response generation. As shown in Figure 7, we observe the significant performance improvements on using ten facts against using three and five facts, while the performance difference between the three and five is marginal. We suggest that this result should be interpreted with the retrieval results on the right side of Figure 7, where about 40% of retrieved subgraphs including the ten different facts contain at least one necessary knowledge, thus the generation performance is boosted according to the improvement in retrieval.

F.2. Discussions on Using Larger PLMs

Notably, we observe that the use of larger Pre-trained Language Models (PLMs) – three times more number of parameters compared to T5-small that we use – does not result in better performance for the knowledge-grounded dialogue task. Specifically, in Table 6, we report the experimental results of selected baselines and our SURGE semi-supervised model with BART-base (Lewis et al., 2020a) as the base PLM. We want to clarify that the BART-base model has 220M parameters, which is about **three times larger** than the number of parameters of the T5-small model (60M).

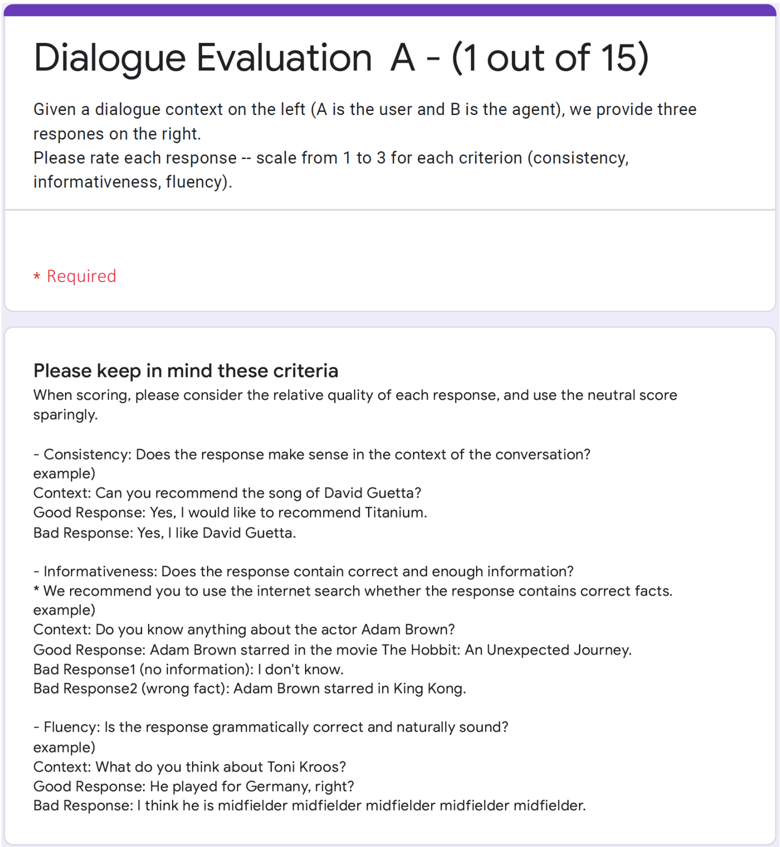
We first observe that BART-base shows decent performance without any knowledge (No Knowledge) compared to the no-knowledge case of T5-small, verifying that the larger PLM generally contains more factual knowledge within its pre-trained parameters. Moreover, BART-base obtains higher scores in the simple word overlap metrics such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004), whose results further confirm that a larger PLM can generate more natural or syntactically better sentences than the smaller one, thanks to its parameter size.

On the other hand, we find that BART-base is less suffered from the irrelevant knowledge issue (i.e., conditioning irrelevant knowledge for the given context when generating responses) than T5-small, therefore, the performance of *Space Efficient Encoding* on KQA is quite high. However, the use of BART-base does not result in significant improvement on the KQA metric for our SURGE framework. Moreover, ours with T5-small shows better performance than ours with BART-base, when the number of facts within the retrieved subgraph is 10: $n = 10$. This result suggests that the quality of the generated response – having relevant knowledge to the given context – might depend on the performance of the subgraph retriever whose goal is to retrieve the context-relevant knowledge, rather than the inherent performance of PLMs.

F.3. Experimental Results on Another Dataset

In the main paper, we only experiment on OpendialKG dataset (Moon et al., 2019), since it is the largest and most realistic public datasets that provides both dialogues across diverse domains and corresponding large-scale Knowledge Graph (KG) (Bollacker et al., 2008). To verify the effectiveness of our SURGE framework, the existence of the large-scale KG and the importance of relevant fact searching is important since we focus on the real-world scenario where the response generation requires the relevant fact acquirement from the large-scale KG.

However, one can raise the question regarding the versatility of our method on other datasets. To alleviate the issue, we conduct additional experiments on another dataset named KOMODIS (Galetzka et al., 2020), which is also KG-based dialogue dataset. Compared to OpendialKG, KOMODIS does not provide the corresponding large-scale KG and most of responses do not require the knowledge. Therefore, we only measure the automatic evaluation to evaluate the performance of each method on KOMODIS dataset. In Table 7, we present the experimental results on the KOMODIS dataset. Results obviously show that our SURGE framework shows superior performance against baselines on the additional dataset. Therefore, we can conclude that our method can generalize to other datasets beyond the opendialKG dataset.



Dialogue Evaluation A - (1 out of 15)

Given a dialogue context on the left (A is the user and B is the agent), we provide three respones on the right.
Please rate each response -- scale from 1 to 3 for each criterion (consistency, informativeness, fluency).

* Required

Please keep in mind these criteria

When scoring, please consider the relative quality of each response, and use the neutral score sparingly.

- Consistency: Does the response make sense in the context of the conversation?
example)
Context: Can you recommend the song of David Guetta?
Good Response: Yes, I would like to recommend Titanium.
Bad Response: Yes, I like David Guetta.

- Informativeness: Does the response contain correct and enough information?
* We recommend you to use the internet search whether the response contains correct facts.
example)
Context: Do you know anything about the actor Adam Brown?
Good Response: Adam Brown starred in the movie The Hobbit: An Unexpected Journey.
Bad Response1 (no information): I don't know.
Bad Response2 (wrong fact): Adam Brown starred in King Kong.

- Fluency: Is the response grammatically correct and naturally sound?
example)
Context: What do you think about Toni Kroos?
Good Response: He played for Germany, right?
Bad Response: I think he is midfielder midfielder midfielder midfielder midfielder.

Figure 8: **Human Evaluation Instructions.** To measure the qualitative performances of the generated responses, annotators are provided with the following instruction on three criteria – consistency, informativeness, and fluency.

G. Human Evaluation

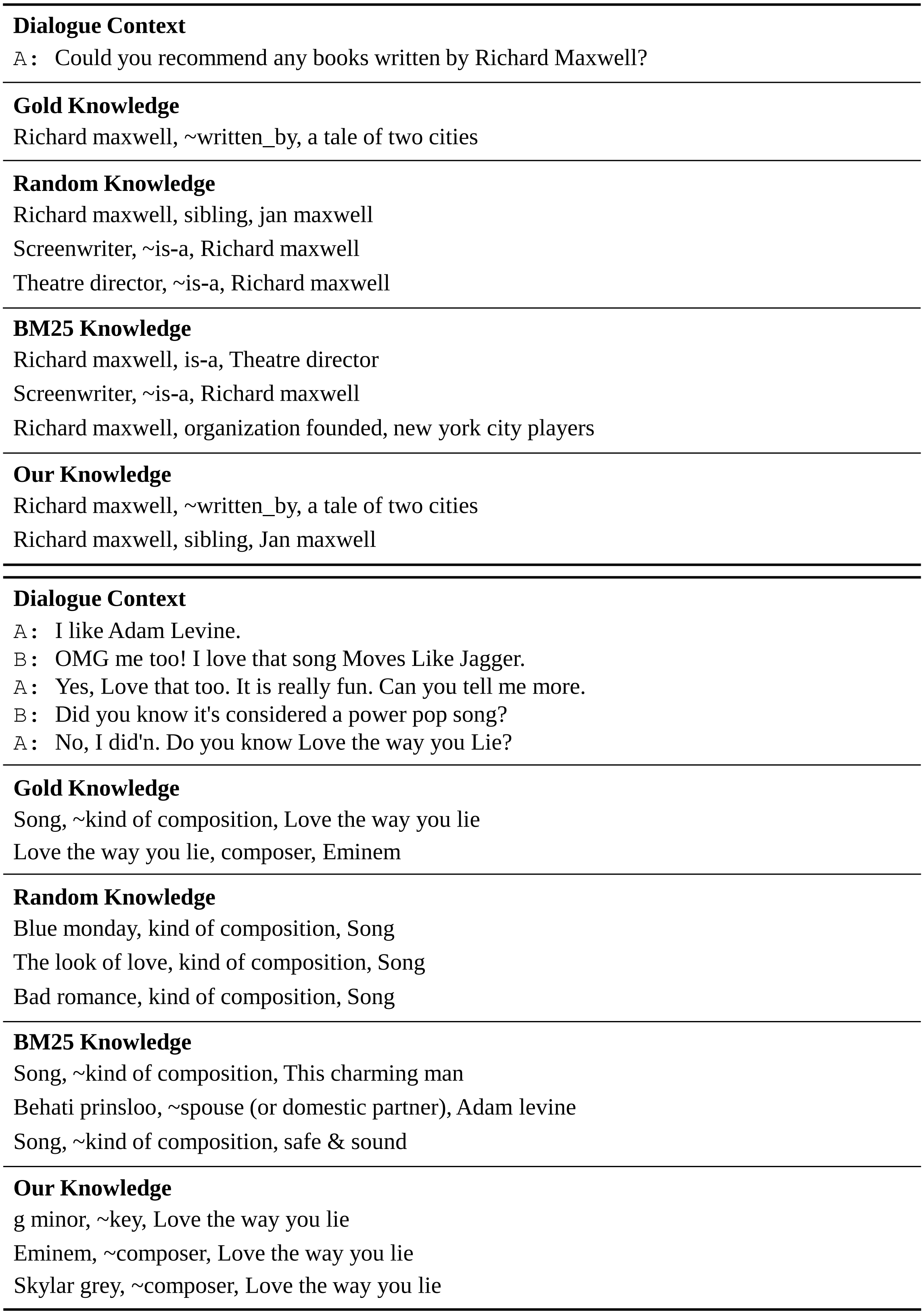
In this section, we describe the details of human evaluation used in Section 5 of the main paper. We request the annotators to evaluate the responses generated from two baselines (i.e., ALL Knowledge and Space Efficient) and our SURGE framework in response to the given dialogue context, according to three criteria – consistency, informativeness, and fluency. Figure 8 is the instructions provided to each annotator. Specifically, regarding the consistency metric, we ask annotators to check whether the generated response makes sense in the context of the conversation. For informativeness, we ask annotators to check whether the response contains correct and enough information, whereby experiment participants are recommended to use the internet search, to check whether the response contains correct facts. In addition to this, we also provide the dialogue-related facts from Freebase as a reference for fact checking for annotators. For fluency, we ask annotators to check whether the response is grammatically correct and naturally sound.

H. Retrieval and Generation Examples

In this section, we provide the examples for knowledge retrieval and response generation, for the given dialogue history.

Retrieval Examples We provide the retrieval examples of various models, such as random retrieval, sparse retrieval and our SURGE models. In particular, in the first (top) example of Figure 9, we are given a dialogue context in regard to books for Richard Maxwell, and baselines including random and BM25 retrievers select the facts associated to the entity Richard Maxwell, which are but irrelevant to the ongoing conversion, for example, (Richard maxwell, is-a Theatre director). Also, as shown in the second (bottom) example of Figure 9, we observe that the simple term-based matching model (i.e., BM25) cannot contextualize the current and previous dialogues, but retrieves the facts associated to frequent words, for example, song, which are less meaningful for the user's question. In contrast to baselines, as our SURGE framework trains a retriever in an end-to-end fashion, it first contextualizes the given dialogue context, and then accurately retrieves relevant knowledge.

Generation Examples In this paragraph, we provide the generation examples from our model. To be specific, we provide the dialogue context along with its corresponding retrieved subgraph and generated response obtained from our SURGE framework. In Figure 10 and Figure 11, we provide the correct examples: our model retrieves a context-relevant subgraph, but also generates a factual response from retrieved knowledge. On the other hand, in Figure 12, we provide the failure cases. In particular, as shown in the first row of Figure 12, the fact in the knowledge graph could be ambiguous or inaccurate, as it defines the release year of the book – Wicked – as both 2008 and 2014. Moreover, we further provide the failure example on retrieval in the second row of Figure 12, where the user asks about the Bourne Legacy, while the dialogue agents retrieve the irrelevant knowledge to the question. Finally, we show the common problem in PLMs in the last row of Figure 12, where the generative model repeats the meaningless words at the end, while the retriever correctly selects the relevant knowledge.



Dialogue Context

A: Could you recommend any books written by Richard Maxwell?

Gold Knowledge

Richard maxwell, ~written_by, a tale of two cities

Random Knowledge

Richard maxwell, sibling, jan maxwell

Screenwriter, ~is-a, Richard maxwell

Theatre director, ~is-a, Richard maxwell

BM25 Knowledge

Richard maxwell, is-a, Theatre director

Screenwriter, ~is-a, Richard maxwell

Richard maxwell, organization founded, new york city players

Our Knowledge

Richard maxwell, ~written_by, a tale of two cities

Richard maxwell, sibling, Jan maxwell

Dialogue Context

A: I like Adam Levine.
B: OMG me too! I love that song Moves Like Jagger.
A: Yes, Love that too. It is really fun. Can you tell me more.
B: Did you know it's considered a power pop song?
A: No, I did'n. Do you know Love the way you Lie?

Gold Knowledge

Song, ~kind of composition, Love the way you lie

Love the way you lie, composer, Eminem

Random Knowledge

Blue monday, kind of composition, Song

The look of love, kind of composition, Song

Bad romance, kind of composition, Song

BM25 Knowledge

Song, ~kind of composition, This charming man

Behati prinsloo, ~spouse (or domestic partner), Adam levine

Song, ~kind of composition, safe & sound

Our Knowledge

g minor, ~key, Love the way you lie

Eminem, ~composer, Love the way you lie

Skylar grey, ~composer, Love the way you lie

Figure 9: Examples of the dialogue history with its corresponding gold knowledge as well as the retrieved knowledge from random retrieval and sparse retrieval baselines and from our SURGE framework. The retrieved fact is represented as the format of (head, relation, tail), where ∼symbol in the front of relation (i.e., ∼relation) in the retrieved knowledge denotes the inverse relation.

Dialogue Context

A: Do you like Rob Marshall? My friends have recommend his films.

Retrieved Knowledge

Nine, has_genre, Drama
Pirates of the Caribbean: On Stranger Tides, directed_by, Rob Marshall
Memoirs of a Geisha, directed_by, Rob Marshall

Generated Response

B: Yes, he wrote Pirates of the Caribbean: On Stranger Tides, Memoirs of a Geisha Nine and many others. Have you seen either of those?

Dialogue Context

A: Are there any works by Chris Cleave, which you can recommend me?
B: Certainly, wrote The Other Hand, Incendiary, and Old. Have you read them?
A: No, I haven't read The Other Hand out of the ones you mentioned. What genre is it?

Retrieved Knowledge

The Other Hand, release_year, 2008
2008, ~release_year, The Other Hand
The Other Hand, has_genre, Literary fiction

Generated Response

B: It's Literary fiction, and was released in 2008.

Dialogue Context

A: Who wrote Of Mice and Men?
B: It was written by John Steinbeck
A: Didn't he also write The Red Pony?
B: Yes He also wrote The Red Pony
A: Can you suggest other titles by him to read?

Retrieved Knowledge

Cannery Row, written_by, John Steinbeck
The Grapes of Wrath, written_by, John Steinbeck
Tortilla Flat, written_by, John Steinbeck

Generated Response

B: Yes he also wrote Tortilla Flat

Figure 10: Examples of the dialogue history with its corresponding retrieved knowledge and generated response from our SURGE framework. The fact is represented as the format of (head, relation, tail), where ∼symbol in the front of relation (i.e., ∼relation) in the retrieved knowledge denotes the inverse relation. In this example, we only provide the correct cases of both retrieval and generation.

Dialogue Context

A: Could you recommend some movies by director Simon Wood?

Retrieved Knowledge

The One That Got Away, written_by, Simon Wood

Simon Wood, ~written_by, The One That Got Away

Author, ~is-a, Simon Wood

Generated Response

B: Simon Wood directed The One That Got Away. Have you seen that?

Dialogue Context

A: I like David McCullough. Could you recommend any books of him?

B: Sure. He wrote The Greater Journey: Americans In Paris. Also, he wrote some documentary and Indie films.

A: Thank you for the information. When was The Greater Journey: Americans In Paris released?

Retrieved Knowledge

The Greater Journey: Americans In Paris, release_year, 2011

2011, ~release_year, The Greater Journey: Americans In Paris

Paris, written_by, Edward Rutherfurd

Generated Response

B: It was released in 2011. It's a great book.

Dialogue Context

A: I like Tony Tanner, what do you think?

Retrieved Knowledge

Sense and Sensibility, written_by, Tony Tanner

Tony Tanner, ~written_by, Mansfield Park

Mansfield Park, written_by, Tony Tanner

Generated Response

B: I think he is a great actor. He starred in Sense and Sensibility and Mansfield Park.

Figure 11: Examples of the dialogue history with its corresponding retrieved knowledge and generated response from our SURGE framework. The fact is represented as the format of (head, relation, tail), where ∼symbol in the front of relation (i.e., ∼relation) in the retrieved knowledge denotes the inverse relation. In this example, we only provide the correct cases of both retrieval and generation.

Dialogue Context

A: Do you know what books Jennifer L. Armentrout has written?

B: She wrote Wicked, White House Kiss and Wait For You. Are you interested in her other books as well?

A: No thanks. What genre is Wicked?

B: Wicked is a Play that won the Pulitzer Prize for Drama. Would you like more information about it?

A: Sure, sounds interesting

Retrieved Knowledge

Wicked, release_year, 2008

2008, ~release_year, Wicked

Wicked, release_year, 2014

Generated Response

B: It is a play written by Jennifer L. Armentrout and released in 2008.

Dialogue Context

A: I like the movie Armageddon.

B: Did you know Armageddon is written by Tony Gilroy?

A: Oh, didn't he also write The Bourne Legacy?

Retrieved Knowledge

Michael Clayton, starred_actors, Tony Gilroy

Michael Clayton, written_by, Tony Gilroy

Proof of Life, written_by, Tony Gilroy

Generated Response

B: Yes, he did. He also wrote Michael Clayton.

Dialogue Context

A: I like Deadpool or any movies like Deadpool.

Retrieved Knowledge

Deadpool, starred_actors, Ryan Reynolds

Deadpool, starred_actors, Stefan Kapicic

Deadpool, starred_actors, Ed Skrein

Generated Response

B: Deadpool is a great movie. Stefan Kapicic starred in it. Stefan Kapicic also starred in The Last Man and The Last Man.

Figure 12: Examples of the dialogue history with its corresponding retrieved knowledge and generated response from our SURGE framework. The fact is represented as the format of (head, relation, tail), where ∼symbol in the front of relation (i.e., ∼relation) in the retrieved knowledge denotes the inverse relation. In this example, we only provide the failure cases due to the problem on data (first row), retrieval (second row), and generation (third row).