CARE: A Clue-guided Assistant for CSRs to Read User Manuals

Weihong Du[12] **Jia Liu**[1] **Zujie Wen**[3]
Dingnan Jin[4] **Hongru Liang**[12*] **Wenqiang Lei**[12]
[1]College of Computer Science, Sichuan University, China
[2]Engineering Research Center of Machine Learning and Industry Intelligence, Ministry of Education, China
[3]Dalian University of Technology, China
[4]University of Electronic Science and Technology, China
duweihong@stu.scu.edu.cn {lj_m, wenzj2010}@163.com
alex2triten@gmail.com
{lianghongru, wenqianglei}@scu.edu.cn

Abstract

It is time-saving to build a reading assistant for customer service representations (CSRs) when reading user manuals, especially information-rich ones. Current solutions don't fit the online custom service scenarios well due to the lack of attention to user questions and possible responses. Hence, we propose to develop a time-saving and careful reading assistant for CSRs, named **CARE**. It can help the CSRs quickly find proper responses from the user manuals via explicit clue chains. Specifically, each of the clue chains is formed by inferring over the user manuals, starting from the question clue aligned with the user question and ending at a possible response. To overcome the shortage of supervised data, we adopt the self-supervised strategy for model learning. The offline experiment shows that **CARE** is efficient in automatically inferring accurate responses from the user manual. The online experiment further demonstrates the superiority of **CARE** to reduce CSRs' reading burden and keep high service quality, in particular with $> 35\%$ decrease in time spent and keeping a > 0.75 ICC score.

1 Introduction

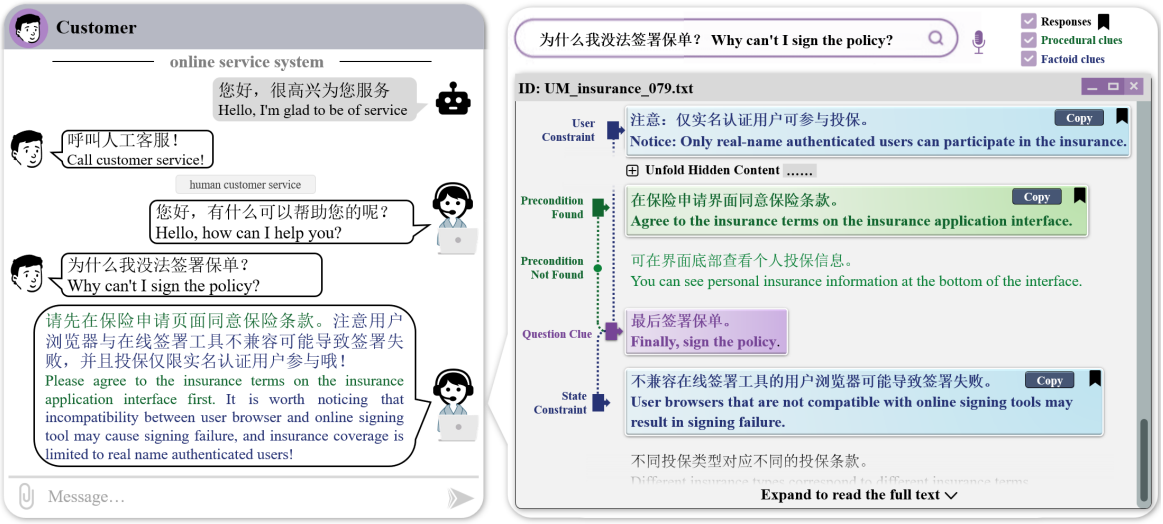
In e-commerce, human services are always involved — once called, a custom service representation (CSR) is required to answer the user's questions based on a related user manual (Zhou et al., 2023). This job is not easy, as the CSR needs to read the information-rich user manual and compose a proper response to the user, as shown in Figure 1 (a). It will be greatly time-saving if we build a reading assistant for CSRs when reading user manuals and seeking possible responses.

Current reading assistants (Badam et al., 2018; Fok et al., 2023) have demonstrated that highlighting special contents (e.g., the co-reference phrases in a document, the "Notice" sections of a document, etc) can reduce people's reading burden when skimming over a whole document. However, these solutions don't work well in online customer services because the highlighted contents have few connections to the question raised by a user or possible responses to the question. There is still a blank of assistants that virtually contribute to saving CSRs' time in reading user manuals for locating proper responses to users quickly.

Hence, we consider building a reading assistant for CSRs that can highlight possible responses, which are expected to be accurate enough to greatly save CSRs' time by copying these responses. Beyond this, we are concerned about its practicability in real-world scenarios — if a CSR accidentally copies the responses wrongly predicted by the assistant, it will risk the service quality of online custom service. This forces us to build a more careful reading assistant that can also explicitly explain how to arrive at these responses from the user question. As a remedy, we propose to explain responses via clue chains, cf., Figure 1 (b). Each of the clue chains starts from the question clue, walks through transitional clues, and ends at a response clue. The transitional clues explain how the response clues connect with the question clue. For example, the transitional clue "You can see ... the interface" indicates that the response clue "Agree to the ... application interface" is two steps before "Finally, sign the policy". Note that, besides steps, the facts binding the steps may also be response clues, e.g., "Notice: Only real-name ... the insurance". As such, the CSRs can check the correctness

* Corresponding author

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10795–10811



(a) Online custom service **(b) Clue-guided assistant**

Figure 1: Illustrations of (a) a log of online custom service, where the user prefers to chat with CSR; and (b) the proposed clue-guided reading assistant (**CARE**), which provides clues explaining how to arrive at proper responses.

of response clues more easily and are less likely to copy improper responses predicted by the model.

We call this Clue-guided Assistant for CSRs to REad user manuals as **CARE**. To get explainable clue chains over unstructured user manuals, we convert them into heterogeneous graphs. As such, the explainable comprehension of user manuals turns to explicit inferring over the graphs. Accordingly, the alignment of the user question and its question clue in the user manual can be modeled as finding the most relevant node to the user question in the graph. This is done with the aid of a matrix projecting the user question to the graph space, which is founded on the heterogeneous graph attention network. After getting the question clue node, we model the inference of response clues as the link prediction task from the question clue node to proper response clues. Further, to overcome the shortage of dedicated supervision data, we adopt the self-supervised training strategy to enhance the reasoning ability of the model from the inherent information of user manuals. The offline experiment on the testing set, derived from a Chinese e-commerce platform, demonstrates the efficiency of **CARE** to infer accurate responses to questions. Besides, the online experiment, deployed to train new CSRs in practice, demonstrates that **CARE** can significantly reduce the reading burden of CSRs (saving more than 35% time) and maintain high service quality (> 0.75 ICC score). We summarize our contributions as follows.

- We call attention to the importance of building a time-saving and careful reading assistant in online custom service. It must be efficient to provide accurate responses and be explainable to guide the CSRs to arrive at these responses.
- We propose to use clue chains to achieve such an assistant and develop **CARE**, which can not only obtain proper responses to users but also explain the responses by inferring over structured representations of user manuals.
- Both offline and online experiments demonstrate the superiority of **CARE** in reducing CSR's reading burden and keeping high service quality.

2 Related Work

Reading Assistant Designing an efficient assistant to reduce the human reading burden is vital in building an enjoyable reading experience for information-rich documents (Badam et al., 2018). A line of work claims that connecting co-reference contents can simplify the reading process (Strobelt et al., 2009; Badam et al., 2018; Pinheiro and Poco, 2022), for example, highlighting all words related to "insurance" in the user manual of Figure 1. Another line of work focuses on marking pre-defined and specific contents in the documents (Yang et al., 2017; Cachola et al., 2020; Fok et al., 2023), for example, highlighting all "Notice" sections in the user manual. Although these solutions can direct the reader's focus to the main idea of a document, they can hardly reduce the CSRs' burden because these highlighted contents have few connections to the user question and possible responses. To fill this blank, we develop a clue-guided assistant that

saves CSRs' time by explicitly explaining how to get accurate responses in the user manual.

Explainable Machine Comprehension Despite growing interest, explainable machine comprehension, which requires the model to expose how to arrive at the final responses/answers given a document, is still in the primary stage (Thayaparan et al., 2020). Therefore, most of the existing works are devoted to constructing explanation-supporting benchmarks in machine comprehension (Khashabi et al., 2018; Inoue et al., 2020; Ho et al., 2020; Cui et al., 2022a). However, these annotated explanations only contribute to factoid-style reasoning (Li et al., 2021; Tan et al., 2021; Zhao et al., 2023). There is still a lack of explanation-supporting datasets for user manuals, which contain both steps and facts. This motivated us to leverage the self-information in user manuals and design a self-supervised strategy for model learning. Specifically, we propose an innovative data construction approach that generates both factual and procedural Q&A pairs from user manuals. These augmented data can greatly enhance the model's ability to answer complex user questions.

Knowledge Graph Question Answering Compared with conventional question answering, knowledge graph question answering has the inherent advantage of forming explanations (i.e., inferring path) leading to the answers (Yasunaga et al., 2021; Zhang et al., 2022). This inspires us to convert raw user manuals into structured graphs. In this way, the comprehension of user manuals becomes explicit inferring over the nodes and edges of the graphs. However, current works only characterize factual knowledge in the knowledge graph (Talmor and Berant, 2018; LAN et al., 2021), overlooking the necessity of procedural knowledge for integrated understanding of user manuals. As a remedy, we represent the user manuals as heterogeneous graphs to support the reasoning of responses to complex user questions with the help of both procedural and factual information.

Heterogeneous Graph Reasoning Reasoning on heterogeneous graphs, which are organized as multiple types of nodes and edges, has been used in various tasks that require inference on complex information (Wang et al., 2019; Bing et al., 2023; Wang et al., 2023). For example, Wang et al. (2021) and Yu and Li (2023) proposed cross-view contrastive learning based reasoning algorithms to conduct in-depth reasoning for node classification, node clustering, graph classification, etc. In line with these works, we believe the heterogeneous graph reasoning techniques will also greatly benefit the deep comprehension of a user manual, which contains both factual and procedural types of information. Despite the big potential, similar efforts are seldom seen in handling user manuals, which also contain multiple types of information. An exception is Liang et al. (2023), which is the only surveyed literature representing user manuals as heterogeneous graphs. However, this work uses manually-defined rules to find potential answers to user questions and thus has trouble answering questions beyond the rules. To this end, in this paper, we contribute the primary attempt to learn a heterogeneous graph reasoning model for user manuals and cope with various user questions through explicit inference on both procedural and factual information over graphs.

3 Methodology

The proposed reading assistant **CARE** is founded on accurate responses to user questions and explicit explanations of these responses. We model it as inferring over the structured representations (i.e., heterogeneous graphs, as shown in Figure 2[1]) of user manuals given user questions. Specifically, we propose a backbone model to infer responses as illustrated in Figure 3. The user question is aligned to the user manual by finding the most related node in the graph, namely the question clue node. Starting from the question clue node, we jointly infer procedural and factual clues over the graph until reach the response clue nodes that can compose proper responses to the user question.

3.1 Heterogeneous Graph Construction

Inspired by Liang et al. (2023), we represent the user manual as a heterogeneous graph, which consists of action nodes, entity nodes, arguments nodes, and their relations, as shown in Figure 2.

The procedures of a user manual are formed as several sequences of action nodes by linking actions in the same procedure with the "Next" relation. For example, the relation from the action node "see" to the action node "sign" is "Next". The "AGT" relation links an action node with its agent (e.g., "User") and the "PAT" relation con-

[1]For convenience and saving spaces, some examples are only presented in English.

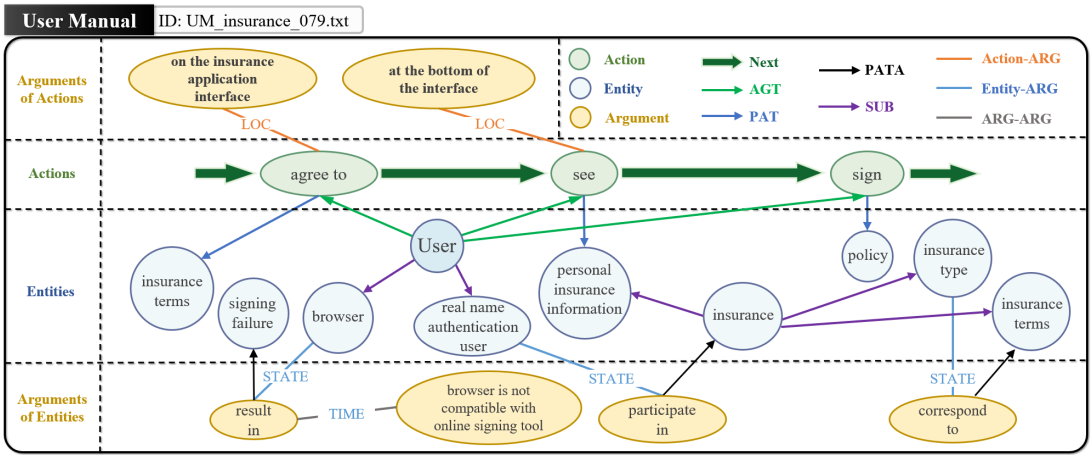


Figure 2: Part of the heterogeneous graph derived from the user manual in Figure 1. Each action on the graph is decorated with corresponding arguments, and the arguments possessed by each entity and the sub-entity relations between different entities are clearly represented by corresponding links. Best viewed in color.

nects each patient (e.g., "policy") with their action. The "Action-ARG" relation links an action node with its arguments, including the time, location, and manner descriptions.

Entity nodes are derived from the concepts[2] in the user manual and linked with their arguments via the "Entity-ARG" relations. For example, the state node "result in" is a changeable state argument of the entity "browser". We allow an entity to have sub-entities and link them via the "SUB" relation. The "PATA" relation links the state node to an entity node, denoting that the entity is affected by the changing of state. The "ARG-ARG" relation describes the augments of an entity's argument, e.g., the time argument of the state argument "result in".

Notably, towards joint inference of procedural and factual knowledge, we summarize the above-mentioned heterogeneous graph into three types of nodes (i.e., action nodes, entity nodes, and argument nodes) and two types of paths ((i.e., the procedural path and the factual path). The procedural path starts from an action node and walks over the graph directed by the "Next" relation. Besides the starting node, all nodes passed by the procedural path are action nodes. What's more, a path constructed from the remaining relations is defined as a factual path.

3.2 Alignment

As illustrated in Figure 3, the first part of the backbone model is aligning the user question to a node in the graph $G = \{V, E\}$, namely the question clue node. Following (Wang et al., 2019), we adopt the heterogeneous graph attention network to encode the action, entity, and argument nodes into vectors.

Specifically, the procedural neighboring nodes set of a node is collected from 1-hop procedural paths, and the factual neighboring nodes set is collected from 1-hop factual paths. The embedding of a node is obtained from two-step fusion — the embeddings of all nodes in the same neighboring nodes set are fused to produce a set embedding; all set embeddings are fused to produce the embedding of the current node through path-level attention. This process can be formulated as

$$\mathbf{n} = \alpha_{\mathbb{P}}\mathbf{s}_{\mathbb{P}} + \alpha_{\mathbb{F}}\mathbf{s}_{\mathbb{F}}, \tag{1}$$

$$\alpha_{\mathbb{P}} = \texttt{softmax}\big[(\sum \mathbf{s}_{\mathbb{P}})/(\sum \mathbf{s}_{\mathbb{P}} + \sum \mathbf{s}_{\mathbb{F}})\big], \tag{2}$$

$$\alpha_{\mathbb{F}} = \texttt{softmax}\big[\sum \mathbf{s}_{\mathbb{F}})/(\sum \mathbf{s}_{\mathbb{P}} + \sum \mathbf{s}_{\mathbb{F}})\big], \tag{3}$$

$$\mathbf{s}_{\mathbb{P}} = \sum\nolimits_{\mathtt{p}\in\mathbb{P}} \mathbf{p}/|\mathbb{P}|, \quad \mathbf{s}_{\mathbb{F}} = \sum\nolimits_{\mathtt{f}\in\mathbb{F}} \mathbf{f}/|\mathbb{F}|, \tag{4}$$

$$\mathbb{P} = \{\mathtt{m}|r(\mathtt{m},\mathtt{n}) = \text{Next}\}, \tag{5}$$

$$\mathbb{F} = \{\mathtt{m}|r(\mathtt{m},\mathtt{n}) \neq \text{Next} \wedge |r(\mathtt{m},\mathtt{n})| \neq 0\}, \tag{6}$$

where $\mathtt{n}$ is the current node, $\mathtt{m}$ denote another node in the graph, $\mathbf{n}$ is the embedding of $\mathtt{n}$, $\mathbb{P}/\mathbb{F}$, $\mathbf{s}_{\mathbb{P}}/\mathbf{s}_{\mathbb{F}}$, $\alpha_{\mathbb{P}}/\alpha_{\mathbb{F}}$, and $\sum \mathbf{s}_{\mathbb{P}}/\sum \mathbf{s}_{\mathbb{F}}$ are the procedural/factual neighboring node set, procedural/factual set embedding, attention weight for procedural/factual information, and the sum of all procedural/factual set embeddings in the graph, respectively. In this way, each node is jointly embedded from both procedural and factual information associated with it.

Further, the user question is encoded as a vector using pre-trained language models. To bridge the semantic gap between a node ($\mathtt{n}$) and the question ($\mathtt{q}$), we learn a matrix $\mathtt{W}$ to project the question embedding into the graph spaces. The alignment process can be formulated as $\arg\max_{\mathtt{n}} \texttt{softmax}(\mathbf{n} \cdot \mathbf{W} \cdot \mathbf{q})$. In other words,

[2]Each user manual has a "User" entity by default.

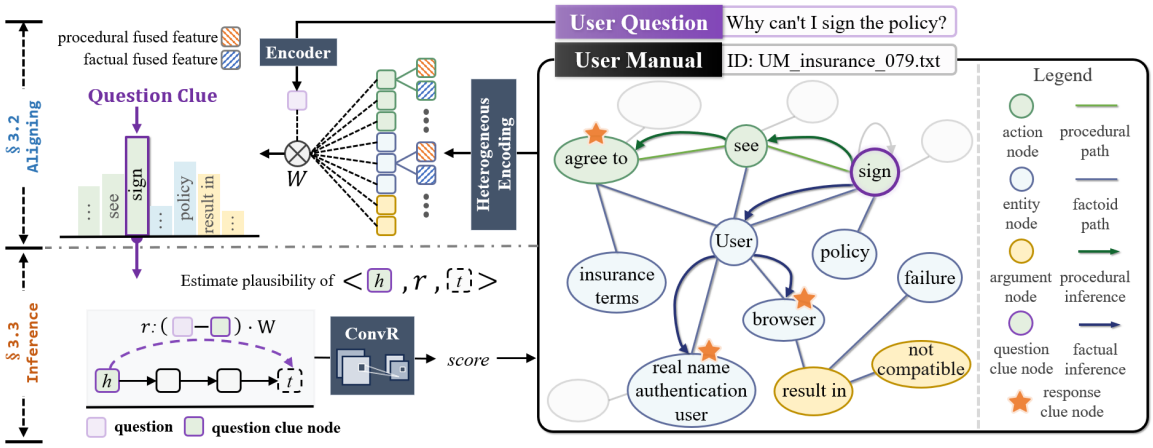


Figure 3: The backbone model of **CARE**, which first aligns the user question to the graph of the user manual and then infers procedural and factoid clues over the graph from the question clue node.

the question clue node is the most similar one to the user question. Notably, although we only align the user question to a single node in the graph, we believe this node is enough to cover all information in the user question. Because the embedding of a node has integrated adjacent arguments information and contains the core concept of the user question which is necessary for further inference. For instance, the action node "sign" has integrated the information of its argument "policy" and can represent the complete action "sign the policy", which is sufficient for subsequent clue inference.

3.3 Inference

We model the inference of a possible response as the link prediction from the question clue node ($\mathtt{n_q}$) to its neighboring nodes. First, we recompose the question clue node as a question clue. For example, the "align" node is combined with its arguments and forms the question clue "Finally, sign the policy" ($\mathtt{Q_c}$). This question clue is encoded as a vector using the same language model as the user question. As such, the required link information (r) can be estimated from $\mathbf{Q_c} - \mathbf{n_q}$. We then project the link information into the graph space for inference, denoted as $\mathbf{r} = \mathbf{W} \cdot (\mathbf{Q_c} - \mathbf{n_q})$. We further add a self-loop edge on $\mathtt{n_q}$ for the case, where the question clue node is also the response clue node.

We then form a triple with the question clue node as the head node ($\mathtt{h}$), a candidate node as the tail node ($\mathtt{t}$), and the link information as their relation ($\mathtt{r}$). The plausibility score of triple $< \mathtt{h}, \mathtt{r}, \mathtt{t} >$ is estimated through the adaptive convolution strategy (Jiang et al., 2019). It splits the link information $\mathbf{r}$ into procedural and factual matrixes, and uses them as filters on the head and tail nodes to produce convolutional features. The plausibility score is computed from the convolutional features.

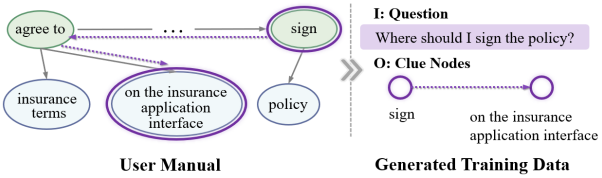


Figure 4: The construction of a sample for self-supervised training.

Towards better coverage of possible responses, we conduct joint inference on both procedural and factual paths with the beam search strategy. It starts from the question clue node ("sign"), walks on the procedural path ("sign" → "see" → "agree to") to find a response clue node and walks on factual paths ("sign" → "User" → "browser" and "sign" → "User" → "real name authentication user") to find the other two response clue nodes (cf., Figure 3,). On either type of path, the inference ends when we meet nodes whose plausibility scores are above a pre-defined threshold δ or the maximum inference depth is reached.

Similar to the recomposition of question clues, the nodes on the path to the response clue node are recomposed as transitional clues, and the response clue node is recomposed as a response clue for highlighting. The paths from the question clue node to the response clue node form the clue chains from the question clue to the response clue, which serve as explicit explanations in our **CARE** assistant.

3.4 Self-supervised Training

Data Construction To overcome the shortage of dedicated annotated data, we adopt the self-supervised strategy to enhance the reasoning ability

of the model only with the inherent information of user manuals. Specifically, we first mask an argument node of an action or entity as the response clue node, then take a node that is not directly connected to it as the question clue node, and generate a natural language question that aligns with the question clue node and can be answered by the masked argument. As we simultaneously generate procedural and factual questions, this enhances the model's ability to answer complex questions that require joint reasoning on both procedures and facts binding them. Figure 4 presents a sample constructed in this way. Specifically, we collect $10,000$ user manuals from a Chinese online e-commerce platform and represent them as heterogeneous graphs. In total, we augment $43,348$ question-answer pairs, each of which is attached with a supporting user manual, for model training.

Objective The objective of the backbone model is to minimize the sum of the binary cross entropy between the predicted clue nodes and the ground truth ones. Note that, during training, we only take the node with the highest plausibility score as the response clue node. During testing, there may be more than one response clue node.

4 Experiments

An ideal reading assistant is expected to have two advantages. First, it should be efficient to highlight accurate responses to user questions. Thus, we design a response clue reasoning experiment to evaluate the ability of the backbone model of **CARE** to infer accurate responses. Second, it should save CSRs' time at low risk by offering clue chains to guide the CSRs to arrive at the responses. We further evaluate **CARE** in the real-world scenario by deploying it to train newly-hired CSRs, namely, the performance test under online deployment.

User manuals are mapped into graphs following Liang et al. (2023) with modifications defined in Sec. A. We use the pre-trained PERT model[3] (Cui et al., 2022b) to embed user questions and initialize nodes embeddings. The threshold δ is set to 0.5. The backbone model of **CARE** is trained on two Nvidia P100 GPUs with the Adam optimizer. The learning rate, beam size, and maximum inference depth are set to $1e-5$, 4, and 3, respectively.

4.1 Response Clue Reasoning

Task Description We construct a real-world testing set from a Chinese online e-commerce platform to test the backbone model's ability to infer accurate responses. It consists of $5,000$ manuals from 35 fields, including finance, healthcare, insurance, electronic products, logistics, etc. Each user manual is attached with its most frequently asked user question and the response to the question extracted from past conversations between CSRs and users. We then hire five senior CSRs to check the reliability of the extracted responses and correct minor errors (e.g., typos). Finally, we obtain a testing set with $4,393$ reliable question-response pairs along with supporting user manuals. The response clue reasoning experiment is based on this testing set — given a user question and its supporting user manual, the model needs to infer the response to the question from the manual.

Baselines We compare the proposed backbone model with five state-of-the-art baselines[4]:

- **PERT** (Cui et al., 2022b), which is a pre-trained machine comprehension model. It takes the concatenation of a user question and its supporting user manual as input and predicts the starting and ending locations of a possible response.
- **Fine-tuned PERT**, which is also the PERT model but fine-tuned on our corpus (cf., Sec. 4).
- **TARA** (Liang et al., 2023), which, similar to our work, also represents user manuals as heterogeneous graphs but uses several manually defined rules to find responses.
- **DRGN** (Zheng and Kordjamshidi, 2022), which is a question-answering model for factoid knowledge graphs and fine-tuned on our training corpus. We form the factoid knowledge graph of a user manual by treating subjects and objects as entities and predicates as relations between them. Given a user question, the DRGN model aligns it to an entity node in the graph and walks on the graph to infer a possible response.
- **ChatGPT** (Ouyang et al., 2022), which is instructed to infer responses from the user manual via prompts. We adopt the Tree of Thought (ToT) (Yao et al., 2023) strategy to infer the response step by step. We first prompt ChatGPT to find actions/entities aligned with the user ques-

[3]We use the implementation released by https://github.com/ymcui/PERT.

[4]For a fair comparison, all models are evaluated on the response with the highest prediction probability.

Table 1: Results of the response clue reasoning experiment. Best performances are highlighted **in bold**.

Method	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Rouge-L	BERTScore
PERT (Cui et al., 2022b)	0.298	0.207	0.162	0.135	0.307	0.251
Fine-tuned PERT	0.433	0.345	0.292	0.254	0.438	0.396
TARA (Liang et al., 2023)	0.397	0.301	0.239	0.195	0.361	0.327
DRGN (Zheng and Kordjamshidi, 2022)	0.405	0.312	0.247	0.206	0.385	0.356
ChatGPT (Ouyang et al., 2022)	0.418	0.322	0.266	0.229	0.418	0.400
CARE	**0.544**	**0.450**	**0.387**	**0.339**	**0.497**	**0.484**

tion, then prompt it to infer the response clues step by step for each aligned action or entity, until reach a possible response to the question.

Metrics Following Cui et al. (2022b) and Zheng and Kordjamshidi (2022), we adopt BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004) and BERTScore (Zhang et al., 2019) as metrics. Both the BLEU scores and Rouge-L score measure the lexical similarity between the predicted response with the gold one. The difference is that the BLEU scores focus more on the precision of predicted responses, but the Rouge-L score focuses more on the recall rate of the predicted responses. Besides, the BERTScore score focuses more on the semantic similarity.

Results The experimental results are shown in Table 1. Compared with the baselines, our backbone model gets the best scores on all metrics. The results demonstrate that our backbone model is able to integrately understand both procedural and factual information in user manuals and infer accurate responses to the user questions. Moreover, the joint inference on both procedural and factual paths in the graphs with the beam search strategy further improves the robustness of the backbone model for inferring the desired responses. Additionally, we have the following observations:

1) Both TARA and our backbone model represent user manuals as heterogeneous graphs. However, the performances of TARA are much worse than ours. This is because compared with the hand-craft rules of TARA, our backbone model is more powerful in making unified inferences of actions and entities and thus has better coverage of various user questions in real-world scenarios. What's more, our backbone model is trained on the constructed training corpus with various generated questions through the self-supervised strategy. This greatly improves the model's generalization ability to cope with complex user questions.

2) DRGN, TARA, and our backbone model all conduct inference on structured graphs of user manuals. DRGN outperforms TARA but still performs much worse than ours. This is because DRGN obtains a more comprehensive understanding of user manuals through self-supervised learning, and TARA is limited by manually designed inference rules. However, DRGN still fails to achieve good performance as it only focuses on the factual information of user manuals. Our backbone model, by contrast, emphasizes both procedural and factual information in manuals and is capable of supporting joint inference to find proper responses.

3) PERT performs the worst among all baselines due to its inability to conduct joint inference on structured data and its lack of knowledge related to the user manuals during the pre-training process. Although both PERT and ChatGPT have not been fine-tuned on the training corpus we constructed, ChatGPT uses a larger volume of pre-trained corpus covering a wide range of fields and thus achieves better performance than PERT.

4) Fine-tuned PERT gains significant performance improvement through learning the inherent information of user manuals, which indicates that our proposed self-supervised learning strategy can greatly improve the model's ability to infer the responses to user questions. However, the fine-tuned PERT model still performs worse than ours due to a lack of joint inference ability. Our backbone model benefits from joint inference on both procedural and factual paths and thus can provide more accurate responses.

5) ChatGPT is good at document comprehension, but its performance is still worse than ours. This is because ChatGPT does not have the explicit reasoning ability, and due to its inherent hallucination issues, it often outputs factually incorrect responses. This poses a significant risk of misleading CSRs in practical scenarios. An ideal reading assistant is expected to save CSRs' time at low risk by offering clue chains to guide the CSRs to arrive at the responses. Our backbone model overcomes this challenge by explicitly performing joint inference on heterogeneous graphs. Therefore, the predicted

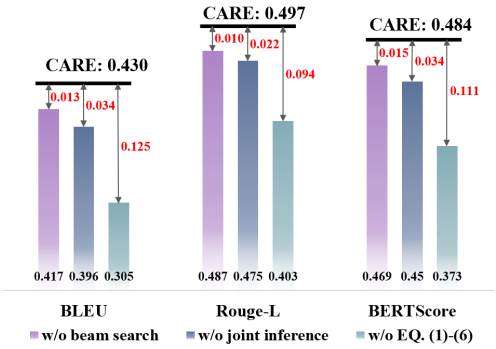


Figure 5: Results of the ablation study

responses of our backbone model are interpretable with inference chains and thus are more practical in real-world applications.

6) Moreover, ChatGPT performs even worse than fine-tuned PERT, this is because the fine-tuned PERT model learns semantic associations between various user questions and corresponding responses during the fine-tuning process, thus can handle more complex user questions. The responses generated by ChatGPT have better semantic coherence, thus slightly outperforming fine-tuned PERT under the BERTScore metric.

Ablation Study The results of ablation experiments are shown in Figure 5, demonstrating the effectiveness of proposed strategies in our backbone model. Note that, the BLEU score here is the average score of BLEU-1, BLEU-2, BLEU-3, and BLEU-4 scores. There is a decline in performance of "w/o beam search" compared with **CARE**. This is because, without the beam search strategy, the accuracy of the response clue node prediction will be heavily affected by the errors accumulated from the previously predicted clue nodes. It can be seen that the performance of "w/o joint inference" drops because, after replacing the joint inference (cf., Sec. 3.3) with direct link prediction to all nodes in the graph, the model loses the ability to reason between procedural and factual information by walking on both procedural and factual paths. We also compare the backbone model of **CARE** with a model "w/o EQ. (1)-(6)". This model encodes the heterogeneous graph in a homogeneous way, where the neighboring node set of a node containing all nodes related to it. There are significant performance decreases of "w/o EQ. (1)-(6)" compared with **CARE**. This is because, without the unified representation defined in EQ. (1)-(6), the model loses the ability to integratedly comprehend the user manuals from both procedural and factual knowledge.

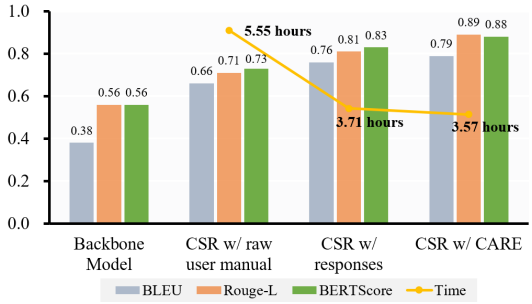


Figure 6: Results with different reading assistants.

4.2 Online Experiment

Task Description We further conduct a live test in the real-world scenario, where newly hired CSRs are trained with the help of reading assistants. Under the between-subjects setting (Charness et al., 2012), we invite three groups of newly hired CSRs, each consisting of two individuals. All of them are asked to answer user questions based on corresponding manuals, with the first group using **CARE**, the second group using the assistant only highlighting response clues, and the third group using the assistant presenting raw user manuals. We also compare these assistants with the backbone model of **CARE**. We select 516 high-frequency access samples from the CSR training bank for testing.

Metrics Besides the BLEU, Rouge-L, and BERTScore scores, we also care about whether it can save time with the help of **CARE**. Hence, we also report the average time spent for a CSR to answer all user questions. To further examine the reliability of human results, we calculate the ICC (Intraclass Correlation Coefficient) (Shrout and Fleiss, 1979) score for each group. The ICC score ranges from 0 to 1, with a high ICC close to 1 indicating high similarity between values from the same group and a low ICC close to zero meaning that values from the same group are not similar. According to Koo and Li (2016), the ICC score, bigger than 0.75, interprets that the values are in good reliability.

Results The results are shown in Figure 6. With the help of **CARE**, the accuracy of the responses provided by CSRs has improved significantly compared to responses given by other CSRs receiving no assistance, and the assistant saves 35.7% the time spent answering user questions for CSRs while keeping a 0.755 ICC score with the gold responses and a 0.801 ICC score with the new CSRs w/ raw user manual. This is because our assistant can guide the CSRs to the proper responses and explain them by showing clue chains, which re-

duces the reading burden of CSRs and improves the efficiency of answering user questions. CSRs receiving no assistance are easily biased by the content at the beginning of the manual due to the heavy reading burden, thus although gaining high consistency, the provided responses miss important response clues in the rest of the user manual. We can also see that the complete display of inferring clue chains can better improve the efficiency and service quality of CSRs, as providing explanations can better guide CSRs to understand the predicted responses and thus they can quickly decide to copy the predicted ones or find the responses themselves. Additionally, relying solely on the responses generated by the backbone model, though answering user questions automatically, gets the worst performances. There is no chance for a CSR to take remedial action for responses wrongly predicted by the model and thus it is too risky to be applied in online custom services.

5 Conclusion

In this paper, we propose a clue-guided assistant **CARE** to help CSRs reduce the burdens of reading user manuals by providing accurate responses and explicitly showing explainable paths about how to arrive at these responses. Specifically, we design a backbone model that aligns user questions to the nodes in the heterogeneous graphs constructed from the manual and infers response clues through joint reasoning along with both procedural and factual paths. In addition, we alleviate the shortage of dedicated annotated data by conducting self-supervised learning on the inherent information of user manuals. Experimental results show that our proposed backbone model can provide accurate response clues for user questions, and the **CARE** assistant can greatly save the time for CSRs to answer questions, thereby improving the efficiency and quality of online customer service. We have deployed our assistant to the novice CSRs training scenario and are preparing to deploy it to the online customer service applications in the future.

6 Ethical Considerations

In this paper, we present a clue-guided reading assistant to help CSRs read user manuals and find proper responses to user questions more quickly. All of the user manuals, questions, and clues involved in our experiments are collected from an e-commerce platform and the collected data in our work does not contain any personal or sensitive information. Therefore, we believe that there are no ethical issues with our work.

7 Limitations

Manually annotating heterogeneous graphs is time-consuming and laborious, so we do not evaluate the adequacy of the constructed heterogeneous graphs. Instead, we evaluate the heterogeneous graphs' effectiveness through their assistance in helping to find accurate responses to user questions and provide explicit explanations of these responses in our experiments. Although the user manuals adopted in our experiments are collected from one online e-commerce platform, they cover various fields, including finance, healthcare, etc. We believe the evaluation results not only demonstrate the effectiveness of **CARE** in the e-commerce CSR setting but show the potential of **CARE** in other CSR settings where CSRs need to frequently read information-rich user manuals. The ideal online experiment is to deploy **CARE** in real service scenarios to test its performance. However, this is not practical as no company is willing to take the danger of making wrong answers, as it may cause poor user experience and may even lose potential customers. As a remedy, we deployed **CARE** in a relative risk-free environment, where newly hired CSRs are trained with the help of reading assistants.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (No. 62206191 and No. 62272330); in part by the Natural Science Foundation of Sichuan (No. 2023NSFSC0473), and in part by the Fundamental Research Funds for the Central Universities (No. 2023SCU12089 and No. YJ202219).

References

Sriram Karthik Badam, Zhicheng Liu, and Niklas Elmqvist. 2018. Elastic documents: Coupling text and tables through contextual visualizations for enhanced document reading. *IEEE transactions on visualization and computer graphics*, 25(1):661–671.

Rui Bing, Guan Yuan, Mu Zhu, Fanrong Meng, Huifang Ma, and Shaojie Qiao. 2023. Heterogeneous graph neural networks analysis: a survey of techniques, evaluations and applications. *Artificial Intelligence Review*, 56(8):8003–8042.

I Cachola, K Lo, A Cohan, and DS Weld. 2020.

Tldr: Extreme summarization of scientific documents. *Findings of EMNLP*.

Gary Charness, Uri Gneezy, and Michael A Kuhn. 2012. Experimental methods: Between-subject and within-subject design. *Journal of economic behavior & organization*, 81(1):1–8.

Wanxiang Che, Yunlong Feng, Libo Qin, and Ting Liu. 2020. N-ltp: A open-source neural chinese language technology platform with pretrained models. *arXiv preprint arXiv:2009.11616*.

Yiming Cui, Ting Liu, Wanxiang Che, Zhigang Chen, and Shijin Wang. 2022a. Expmrc: explainability evaluation for machine reading comprehension. *Heliyon*, 8(4).

Yiming Cui, Ziqing Yang, and Ting Liu. 2022b. Pert: Pre-training bert with permuted language model. *arXiv preprint arXiv:2203.06906*.

Raymond Fok, Hita Kambhamettu, Luca Soldaini, Jonathan Bragg, Kyle Lo, Marti Hearst, Andrew Head, and Daniel S Weld. 2023. Scim: Intelligent skimming support for scientific papers. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, pages 476–490.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. 2020. R4c: A benchmark for evaluating rc systems to get the right answer for the right reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750.

Xiaotian Jiang, Quan Wang, and Bin Wang. 2019. Adaptive convolution for multi-relational learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 978–987.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Terry K Koo and Mae Y Li. 2016. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of chiropractic medicine*, 15(2):155–163.

Yunshi LAN, Gaole HE, Jinhao JIANG, Jing JIANG, Wayne Xin ZHAO, and Ji-Rong WEN. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. IJCAI.

Ronghan Li, Lifang Wang, Shengli Wang, and Zejun Jiang. 2021. Asynchronous multi-grained graph network for interpretable multi-hop reading comprehension. In *IJCAI*, pages 3857–3863.

Hongru Liang, Jia Liu, Weihong Du, Dingnan Jin, Wenqiang Lei, Zujie Wen, and Jiancheng Lv. 2023. Knowing-how & knowing-that: A new task for machine comprehension of user manuals. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10550–10564.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ye Liu, Semih Yavuz, Rui Meng, Dragomir Radev, Caiming Xiong, and Yingbo Zhou. 2022. Uni-parser: Unified semantic parser for question answering on knowledge base and database. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8858–8869.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Joao Pinheiro and Jorge Poco. 2022. Charttext: Linking text with charts in documents. *arXiv preprint arXiv:2201.05043*.

Wei Shi and Vera Demberg. 2019. Next sentence prediction helps implicit discourse relation classification within and across domains. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 5790–5796.

Patrick E Shrout and Joseph L Fleiss. 1979. Intraclass correlations: uses in assessing rater reliability. *Psychological bulletin*, 86(2):420.

Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel A Keim, and Oliver Deussen. 2009. Document cards: A top trumps visualization for documents. *IEEE transactions on visualization and computer graphics*, 15(6):1145–1152.

Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. 2020. Sparqa: skeleton-based semantic parsing for complex questions over knowledge bases. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8952–8959.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Hongye Tan, Xiaoyue Wang, Yu Ji, Ru Li, Xiaoli Li, Zhiwei Hu, Yunxiao Zhao, and Xiaoqi Han. 2021. Gcrc: A new challenging mrc dataset from gaokao chinese for explainable evaluation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1319–1330.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2020. A survey on explainability in machine reading comprehension. *arXiv preprint arXiv:2010.00389*.

Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and S Yu Philip. 2023. A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. *IEEE Transactions on Big Data*, 9(02):415–436.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Peng Cui, Philip S. Yu, and Yanfang Ye. 2019. Heterogeneous graph attention network. *The World Wide Web Conference*.

Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1726–1736.

Qian Yang, Gerard de Melo, Yong Cheng, and Sen Wang. 2017. Hitext: Text reading with dynamic salience marking. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 311–319.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546.

Jianxiang Yu and Xiang Li. 2023. Heterogeneous graph contrastive learning with meta-path contexts and weighted negative samples. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 37–45. SIAM.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yao Zhang, Peiyao Li, Hongru Liang, Adam Jatowt, and Zhenglu Yang. 2021. Fact-tree reasoning for n-ary question answering over knowledge graphs. *arXiv preprint arXiv:2108.08297*.

Yao Zhang, Peiyao Li, Hongru Liang, Adam Jatowt, and Zhenglu Yang. 2022. Fact-tree reasoning for n-ary question answering over knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 788–802.

Wenting Zhao, Justin T Chiu, Claire Cardie, and Alexander M Rush. 2023. Hop, union, generate: Explainable multi-hop reasoning without rationale supervision. *arXiv e-prints*, pages arXiv–2305.

Chen Zheng and Parisa Kordjamshidi. 2022. Dynamic relevance graph network for knowledge-aware question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1357–1366.

Qi Zhou, Bin Li, Lei Han, and Min Jou. 2023. Talking to a bot or a wall? how chatbots vs. human agents affect anticipated communication quality. *Computers in Human Behavior*, 143:107674.

A Details of the Constructed Graphs

Here we describe the constructed graphs of user manuals and explain how can the model infer the desired clues of user questions on the graph through joint reasoning on both procedural and factual information.

A.1 Graph Structure of User Manuals

We construct user manuals into graphs as shown in Figure 7. The graph consists of three kinds of nodes (i.e. action nodes, entity nodes, and argument nodes) and eight kinds of relations (i.e. Next, AGT, PAT, PATA, SUB, Action-ARG, Entity-ARG, and ARG-ARG). Firstly, the procedures containing a list of action nodes are constructed by linking actions belonging to the same procedure with the "Next" relation. We use the Next Sentence Prediction(NSP) of the pre-trained model (Shi and Demberg, 2019) to determine whether two actions belong to the same procedure and construct all the procedures. To establish connections between agents and actions, we employ the "AGT" relation, while the "PAT" relation is utilized to connect patients with their respective actions. And the arguments of each action are linked through the "Action-ARG" relation, including the time, location, and manner description of the corresponding action. Secondly, we construct entity nodes and link their corresponding arguments through the "Entity-ARG" relation. The sub-entity of a certain entity is connected to it through a "SUB" relation. We also link the state of the entity with corresponding entities affected by it through the "PATA" relation and link the arguments associated with other entities' arguments through the "ARG-ARG" relation. Finally, We adopt character level matching to confirm the same entities and arguments references and fuse the same reference nodes into one node to avoid duplicate reference nodes of the constructed graphs.

To support the heterogeneous inference of our proposed model, we retain the original three node types (e.g.) and classify edge types into procedural and factual links. The former only includes the Next relationship in the graph, while the remaining relationships belong to factual link

A.2 Response Clues Inference

We construct the graphs of user manuals and adopt heterogeneous desired clues inference of user questions. With the constructed graph, our proposed model can perform joint reasoning on both procedural and factual information for sufficient and robust inference. As shown in Figure 7, to answer the user question "Why can't I sign the policy?", the model first needs to identify the core action "sign" that the user question is targeting as the question aligned node for subsequent inference of the desired clues. It should be noted that the embedding of each node on the graph has integrated adjacent arguments information that is necessary for further inference. As shown in Figure 8, the action node "sign" has integrated the information of its argument "User" and "policy", and can represent the complete action "User sign the policy", which is sufficient for subsequent clue inference. Starting from the question-aligned node, the model conducts in-depth reasoning on the graph of the manual to find potential clues to the user question. Through joint reasoning on both procedural and factual information, the model first refers to the previous action "agree to" of the aligned node through procedural meta-path based inference and finds the potential clue for the failure of the action "sign", that is the user did not complete the prerequisite operation "agreeing to the insurance terms on the insurance application interface", resulting in the inability to "sign the policy". Then the model further infers to the restrictions "Only real name authenticated users can participate in the insurance" associated with "user" node's sub-entity "real name authentication user" and "User browsers that are not compatible with online signing tools may result in signing failure" associated with "user" node's sub-entity "user browser" through factual meta-path based inference, which may also result in the failure of the action "sign". Through the constructed graph, our proposed model can perform joint reasoning on both procedural and factual information to find potential clues and guide CSRs to reach the responses to user questions.

B Analysis of the Augmented Data

We demonstrate the diversity of augmented data by calculating the action and patient distribution of questions. After syntactic parsing[5], we get $30,670$ questions holding "action-patients" structure. Specifically, the actions are verbs directly connected to the root in parsing trees and their patients are noun objects directly connected to the actions. We present the top-10 actions and their

[5]We use the syntactic parsing tool (Che et al., 2020).

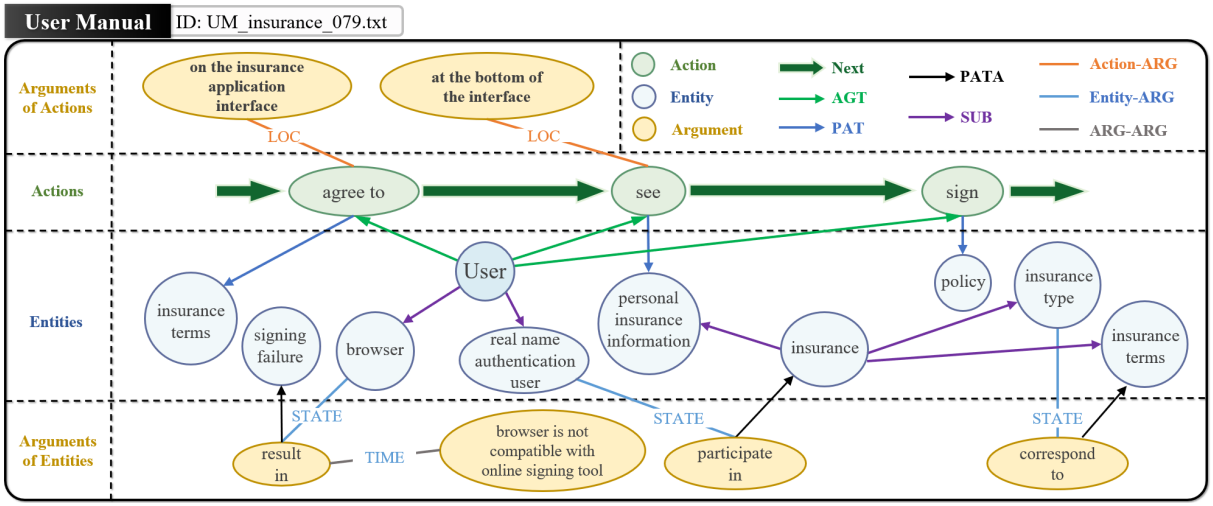


Figure 7: An example of the constructed graph.

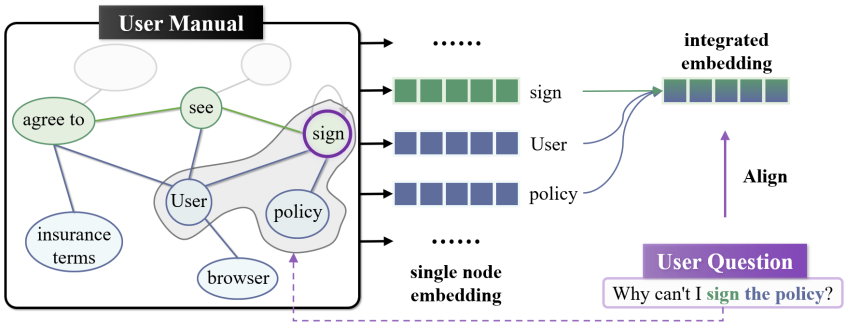


Figure 8: An example of the integrated information of node "sign".

top-4 patients of the generated questions in Figure 9. It can be seen that the generated questions cover a wide range of actions, with a total of 4,010 samples displayed, accounting for only 9.25% of all generated questions and 13.07% of questions with the "action-patient" structure. These generated procedural and factual questions enhance the model's reasoning ability of response clues for various questions raised by users in real-world scenarios and can improve the generalization ability of the model.

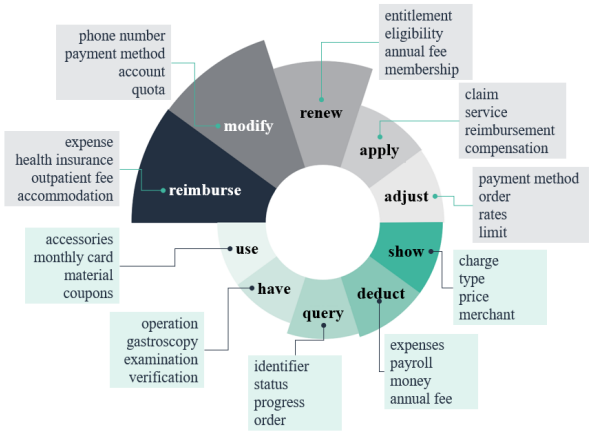


Figure 9: The top-10 actions and their top-4 patients of the questions in the augmented data.

C Computation Details of the Plausibility Score of Triple

We model the inference of a possible response as the link prediction from the question clue node ($\mathtt{n_q}$) to its neighboring nodes. The question clue is encoded as a vector using the same language model as the user question and the required link information (r) can be estimated from $\mathbf{Q_c} - \mathbf{n_q}$. We project the link information into the graph space for inference, denoted as $\mathbf{r} = \mathbf{W} \cdot (\mathbf{Q_c} - \mathbf{n_q})$. Then we form a triple with the question clue node as the head node ($\mathtt{h}$), a candidate node as the tail node ($\mathtt{t}$), and the link information as their relation ($\mathtt{r}$). The plausibility score of triple $< \mathtt{h}, \mathtt{r}, \mathtt{t} >$ is estimated through the adaptive convolution strategy (Jiang et al., 2019). It splits the link information $\mathbf{r}$ into procedural and factual matrixes, and uses them as filters on the head and tail nodes to produce convolutional features. The plausibility score is computed from the convolutional features.

We first reshape question clue node feature $\mathbf{n_q} \in$

$\mathbb{R}^{d_n}$ into 2D matrix $\mathbf{S} \in \mathbb{R}^{d_n^h \times d_n^w}$ and feed it into CNN layer. Here $d_n = d_n^h d_n^w$. To handle complex questions require multiple clues, we split relation feature $\mathbf{r} \in \mathbb{R}^{d_r}$ into same size pieces $\mathbf{r}^{(1)}, ..., \mathbf{r}^{(c)}$ (each $\mathbf{r}^{(l)} \in \mathbb{R}^{d_r/c}$) and reshape each $\mathbf{r}^{(l)}$ into 2D filter $\mathbf{R}^{(l)} \in \mathbb{R}^{h \times w}$. Here c denotes the number of filters, h denotes the height of each filter and w denotes the width of each filter. After that, multiple relation-specific convolutions are conducted between matrix $\mathbf{S}$ and each filter $\mathbf{R}^{(l)}$:

$$C^l = g(\mathbf{S} * \mathbf{R}^{(l)}) \tag{7}$$

where C^l denotes a feature map derived from filter $\mathbf{R}^l$, g denotes the ReLU activation function and $*$ denotes convolution operation. To estimate the plausibility score of triple $< \mathtt{h}, \mathtt{r}, \mathtt{t} >$, we concatenate all feature maps $C^1, ..., C^c$ into single feature $\mathbf{c}$ and project it into $\mathbb{R}^{d_n}$ through fully-connected layer. The plausibility score of the triple can be calculated as follows:

$$score(< \mathtt{h}, \mathtt{r}, \mathtt{t} >) = \mathtt{f}(\mathtt{g}(\mathbf{W_s} \cdot \mathbf{c} + \mathbf{b_s})^\top \cdot \mathbf{n_q}) \tag{8}$$

where $\mathtt{f}$ denotes the sigmoid function, $\mathtt{g}$ denotes the ReLU activation function, $\mathbf{W}_s$ and $\mathbf{b}_s$ are fully-connected layer parameters.

D Details of Adopted Metrics

The BLEU (Papineni et al., 2002) metric is used to evaluate the quality of generated text by measuring the n-gram overlap between the candidate and the reference texts, which is calculated as:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log(p_n)\right) \tag{9}$$

where BP is the brevity penalty, N is the maximum n-gram size, w_n is the weight for each n-gram size and p_n is the precision for each n-gram size.

The ROUGE-L (Lin, 2004) metric is used to evaluate the quality of generated text by measuring the longest common sub-sequence between the candidate and the reference texts, which is calculated as:

$$R_{LCS} = \frac{\text{LCS}(r, c)}{\text{len}(r)} \tag{10}$$

$$P_{LCS} = \frac{\text{LCS}(r, c)}{\text{len}(c)} \tag{11}$$

$$\text{ROUGE-L} = \frac{(1 + \beta^2) R_{LCS} P_{LCS}}{R_{LCS} + \beta^2 P_{LCS}} \tag{12}$$

where $\text{LCS}(r, c)$ is the length of the longest common sub-sequence between reference text r and candidate text c, and $\text{len}(r)$ is the length of the reference text r, $\text{len}(c)$ is the length of the candidate text c, R_{LCS} represents recall rate, P_{LCS} represents precision rate, β is used to adjust the attention to precision and recall rates. Due to β being set to a large number, ROUGE-L more considers recall rate.

The BERTScore (Zhang et al., 2019) metric is used to evaluate the quality of generated text by comparing it to reference text using contextualized embeddings generated by the BERT model. It computes the similarity between the contextualized embeddings of the candidate and reference text, and aggregates the scores to provide an overall BERTScore.

The ICC (Intraclass Correlation Coefficient) (Shrout and Fleiss, 1979) score ranges from 0 to 1, with a high ICC close to 1 indicating high similarity between the evaluation values from the same group, which is calculated as:

$$\text{ICC} = \frac{\text{MS}_{\text{between}} - \text{MS}_{\text{within}}}{\text{MS}_{\text{between}} + (k-1) \times \text{MS}_{\text{within}}} \tag{13}$$

where $\text{MS}_{\text{between}}$ is the mean square for between groups variability, $\text{MS}_{\text{within}}$ is the mean square for within groups variability and k is the number of groups.

E Error Analysis

As the focus of our work lies in the utilization of heterogeneous graphs, we will briefly discuss the error analysis of graph constructing of our method here. For the whole pipeline of our proposed approach, only the process of heterogeneous graphs constructing from the user manuals relies on the semantic parsing results of our chosen NLP tool LTP [6], and the parsing errors may have an impact on the performance of our proposed approach (e.g., identify wrong argument node). However, after investigating the analysis of other parsing-enhanced approaches (Sun et al., 2020; Zhang et al., 2021; Liu et al., 2022), we conjecture these errors will not make a big difference in the model performances

[6] https://ltp.ai/

and corresponding conclusions. Furthermore, we believe that as fundamental NLP components advance, the influence of these errors on subsequent work will even be smaller.

F Discussion of the Used Artifacts

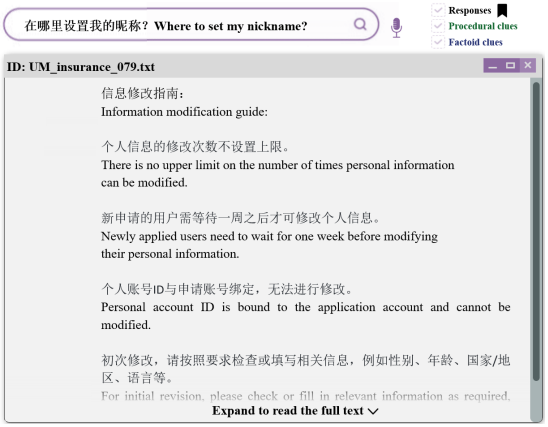
All the codes, data and models used in this paper follow the settings of the original work, and we have listed the citations of all the artifacts we used in References section. All of the user manuals, questions, and clues involved in our experiments are collected from a Chinese online e-commerce platform. All the collected data in our work has been manually desensitized and does not contain any personal or sensitive information, and all the data we use is for research purpose only.

G Case Study

Here we list some cases to demonstrate the superiority of our proposed assistant compared with other assistants. As shown in Figure 10, for the question "Where to set my nickname?", the Co-reference Contents Highlighting approach 10(a) highlights all the spans "nickname" in the manual and the Marking Specific Content approach 10(b) highlights the "kind tips" section, both of them are nothing to do with the user question. Finally, our proposed assistant **CARE** 10(e) successfully locates the response "personal profile page" through in-depth inference on the procedural path. In more complex scenarios, as shown in Figure 11, there is an inconsistency between the user question "Why can't I apply for an international account?" and corresponding manual, that is, specific action failure is not described directly in the user manual. The Co-reference Contents Highlighting approach 11(a) can still only highlight all the spans "Product Service" which are useless for answering the question, and the Marking Specific Content approach 11(b) even fails to match any spans. Unlike other assistants, our **CARE** assistant 11(e) can conduct joint reference on both procedural and factual paths and successfully infers the procedural clue and factual clue for the failure of action "apply for an international account" mentioned in the question. This demonstrates that our proposed assistant can conduct in-depth inference on constructed graphs of manuals and is able to sufficiently locate potential responses through joint inference, further supporting CSRs in handling more complex user questions.

(a) Co-reference Contents Highlighting

(b) Highlighting Specific Content

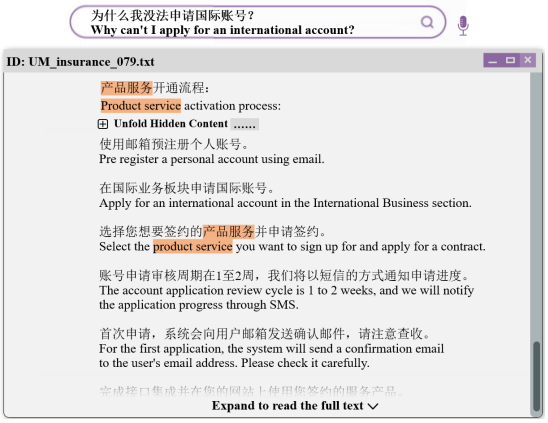


(c) Raw User Manual

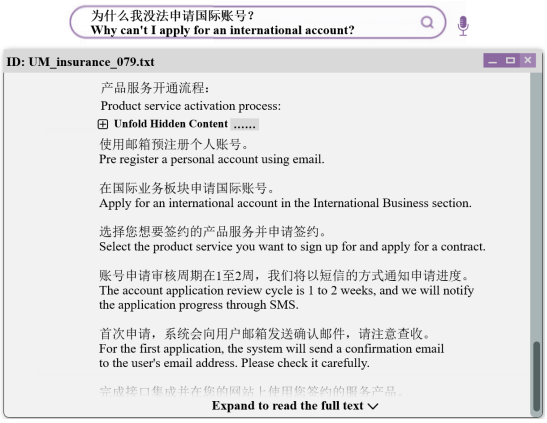
(d) Response Only

(e) Clue-guided Assistant

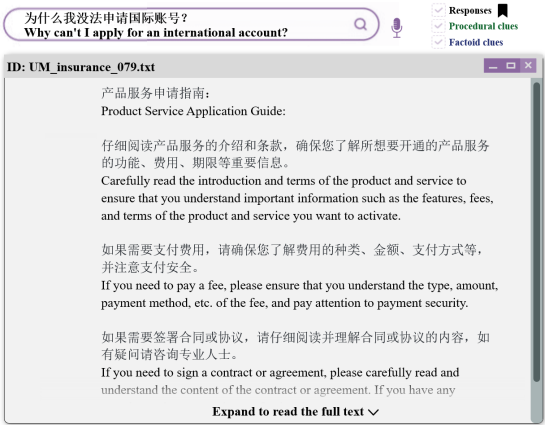
Figure 10: An case of clues provided by different assistants for user question.



(a) Co-reference Contents Highlighting



(b) Highlighting Specific Content



(c) Raw User Manual

(d) Response Only

(e) Clue-guided Assistant

Figure 11: An case of clues provided by different assistants for inconsistent user questions.