Universiteit Leiden

Master Computer Science

Rethinking conversational recommender systems: A reproducibility investigation

Name:	Shupei Li
Student ID:	s3430863
Date:	01/11/2024
Specialisation:	Data Science
1st supervisor:	Zhaochun Ren
2nd supervisor:	Suzan Verberne

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Conversational recommender systems (CRSs) are emerging research topics that benefit from both traditional recommender systems and dialogue systems. Unlike traditional recommender systems that analyze user features from historical data, CRSs directly interact with users to capture their current preferences within conversations. Most existing CRSs leverage external knowledge to enhance system performance, with knowledge graphs being the most commonly used external information material. However, few studies have quantitatively analyzed the impact of knowledge graphs on CRS performance. Additionally, several previous studies have only reported recall and inter-distinct scores, making it more difficult to comprehensively analyze how changes to knowledge graphs influence CRS effectiveness.

In this thesis, we conduct a reproducibility investigation on the role of knowledge graphs in CRSs. We first explore how different amounts of external knowledge affect system performance on recommendation and conversation tasks. Our main finding is that the optimal knowledge graph settings are determined by both the dataset and the metrics. Then, we test CRS performance in the scenario of incomplete knowledge graphs. We find that incomplete knowledge graphs do not necessarily impair model performance. Our experiments cover four representative CRSs, whose performance is evaluated using widely applied recommendation and textual generation metrics. This study addresses gaps in previous research related to knowledge subgraph construction and configuration. Experimental results show that optimizing knowledge graph settings can enhance model performance while also reducing training time.

Contents

Chapter 1

Introduction

Conversational recommender systems (CRSs) have received increasing attention in recent years. The concept of CRSs is inspired by both the traditional recommender systems and the dialogue systems. Traditional recommender systems infer user preferences by analyzing historical data. Although they have been successfully applied in many real-world scenarios such as e-commerce, entertainment, and medical service [36], these type of recommender systems have two major weaknesses. First, the interpretability of most of the recommendation results is still poor, especially for the outputs of deep learning-based methods. The main reason is that deep learning-based methods usually use complex non-linear functions to represent the underlying patterns. From the perspective of system users, the whole recommender system works like a 'black box'. It is difficult to understand why a user prefers a recommended item in most cases [53]. The lack of interpretability hinders the improvement of the model to provide more personalized recommendation services. Second, traditional recommender systems struggle to capture shifts in user preferences in real-time, as they rely heavily on historical data for inference [8].

Unlike traditional recommender systems, CRSs adopt a more proactive recommendation strategy. Specifically, they directly interact with users to collect information about user preferences. This is achieved through a conversational module for interaction and a recommendation module for preference inference [14]. According to the dialogue strategy, CRSs can be divided into two categories, attribute-aware CRSs and topic-guided CRSs [34]. Attribute-aware CRSs ask users to specify their requirements through a series of questions regarding product attributes. In contrast, topic-guided CRSs simulate human-like conversations that occur in real world scenarios with more natural responses, seamlessly transitioning between casual dialogue and recommendation-focused interactions. Our project within this thesis mainly focuses on topic-guided CRSs.

CRSs have two main development directions. One is introducing external knowledge such as knowledge graphs and user reviews to reduce the uncertainty. The other is modifying the model architecture to improve model performance. Efforts in these two directions are complementary to each other.

We argue that effectively utilizing external knowledge is crucial for improving CRS performance. However, as far as we know, only few existing studies have investigated the impact of the different amounts of external knowledge on CRSs performance. Research that focuses on the proposal of novel CRSs rarely reports detailed information about the external knowledge used in the model, for example, the method of subgraph extraction from knowledge databases and the version of knowledge graphs. This leads to the possibility that the amount of external knowledge used in baselines may be different in different studies, which introduces bias in model performance comparison. In addition, there are no unified metrics for the evaluation of CRSs performance. Reported metrics vary across studies. Even for the same metric, the code implementation of different studies may adopt different calculation methods. As a result, it is difficult to obtain a comprehensive performance comparison of different CRSs on commonly used metrics. To bridge these research gaps, we conduct a reproducibility study that explores how different amounts of external knowledge affect the performance of CRSs. We evaluate the system performance on several widely-used metrics with the most commonly used implementations to ensure a fair comparison.

We examine four representative CRSs in this project: KBRD [4], KGSF [59], UniCRS [46], and C^2-CRS [61]. KBRD is the first attempt at incorporating encyclopedic knowledge graphs into the model. KGSF further explores the idea of leveraging external knowledge graphs. It uses not only encyclopedic knowledge graphs but also semantic network-based knowledge graphs. Besides, it also creates the mutual information maximization mechanism to align embedding spaces. C^2-CRS leverages both knowledge graphs and user reviews. It designs a contrastive learning framework to achieve embedding alignment. UniCRS is an attempt to unify the training of recommendation and conversation modules by utilizing pretrained large language models. It is worth mentioning that UniCRS also uses encyclopedic knowledge graphs.

The external knowledge in this study mainly refers to strcutrued knowledge graphs due to their extensive usage in CRSs. The primary challenge we encounter in the project is how to quantify the amount of information brought by external knowledge graphs on a specific dataset. Previous studies did not explain how they extract dataset-relevant subgraphs when introducing external knowledge graphs. However, we believe that the amount of information a knowledge graph brings to the model is influenced by many

factors, such as the subgraph size, its relevance to the data, etc. Our solution is inspired by the user preference propagation process described in RippleNet [44]. The basic assumption is that the influence of user preferences will propagate outward layer by layer like ripples in the knowledge graphs. We define the seed set as entities that appear in the conversational text of the dataset. Then, we control the amount of external knowledge by including the one-hop, two-hop, and three-hop neighbors of the seed on the knowledge graph. Accordingly, we state the first research question as follows:

RQ1 How do amounts of external knowledge affect the performance of CRSs when we include one-hop, two-hop, and three-hop neighbors of entities?

In real-world scenarios, the amount of external knowledge may fluctuate due to the incompleteness of knowledge graphs. Thus, exploring how CRSs perform when knowledge graphs are incomplete is a worthwhile direction. We simulate this by randomly removing a percentage of edges from the graph, leading to our second research question:

RQ2 How does the incompleteness of knowledge graphs affect the performance of CRSs when we randomly remove 10%, 30%, and 50% of edges?

To answer the above research questions, we conduct a series of experiments on three open-source datasets. Contributions of this thesis are summarized as follows:

- We evaluate CRSs in terms of multiple commonly used metrics and follow the standard metric calculation method, thereby providing a comprehensive picture of the model performance.
- By investigating the impact of amounts of knowledge graphs on CRSs performance, we find that extracting larger subgraphs does not necessarily improve system performance. Moreover, the optimal subgraph size varies depending on different evaluation metrics. The best configuration of external knowledge graphs requires a comprehensive consideration of the dataset used and the metrics prioritized.
- We examine the robustness of CRSs when the knowledge graph is incomplete. We find that using incomplete external knowledge graphs does not necessarily lead to poorer model performance. However, our experiments do not reveal a consistent pattern regarding the impact of the degree of incompleteness on model performance.

The rest of the thesis is organized as follows. Chapter 2 reviews related works in rec-

ommender systems, dialogue systems, and conversational recommender systems. Chapter 3 introduces the public datasets and knowledge graphs used in this project. In Chapter 4, we describe the process of constructing subgraphs to control the amount of external knowledge in detail. Chapter 5 summarizes our experimental set-up and presents results. Chapter 6 discusses our experimental results. Finally, Chapter 7 concludes this thesis.

Chapter 2

Related work

2.1 Recommender systems

Recommender systems have been widely used in various fields of the Internet industry, such as e-commerce, social media, streaming services, etc. The applictaion of recommender systems creates a twofolded benefit. On one hand, customers are able to acquire the information they are interested in efficiently under the impact of the information overload problem. Internet companies, on the other hand, increases their business profits by continuously optimizing user experience and attracting more users.

Early recommender systems are rooted in traditional statistical and machine learning models. Collaborative filtering is one of the representative models. Research on collaborative filtering algorithms dates back to the 1990s when Xerox company tried to develop an algorithm based on historical data to filter emails that users are not interested in [9]. Co-occurrence matrix and similarity estimation are the core of collaborative filtering. Rows and columns of the co-occurrence matrix represent the existing users and items in the database, respectively. The vanilla collaborative filtering algorithm, also known as user collaborative filtering (UserCF), predicts customer's preferences for a new product by analyzing the choices of similar users. Commonly used similarity metrics include cosine similarity, Pearson correlation coefficient, constrained Pearson correlation, and Spearman rank correlation. After filtering out the top K most similar users, UserCF will generate the final results according to some reranking functions, such as the weighted average of the user similarity. The intuition behind UserCF's results is that users tend to choose the same products within a small circle of like-minded people. Despite the social features and the potential of quickly detecting the hottest trends, UserCF suffers

from the scalability problem [7]. Besides, users's historical data is usually sparse in real business scenarios, which further reduces the algorithm's accuracy. Item collaborative filtering (ItemCF) is an alternative to UserCF. It computes the item similarity and generates results based on positive feedback from historical data. If the user's interests are stable over a period of time, ItemCF can overcome the shortcomings of UserCF to a certain extent [26].

Matrix factorization and logistic regression are also once popular recommendation algorithms. Compared to collaborative filtering, matrix factorization introduces the usage of latent variables to increase the generalization capability. The core of matrix factorization is decomposing the co-occurrence matrix into the user matrix and the item matrix. There are multiple choices to accomplish matrix factorization, for example, singular value decomposition and stochastic gradient descent method. Improvement directions based on the naive matrix factorization include adding regularization terms and biases. Practical experience during the 2006 Netflix prize competition [18] shows that matrix factorization models significantly outperform classic nearest-neighbor models on large-scale datasets. Logistic regression is a famous statistical model. It maps inputs from a high-dimensional space to a low-dimensional space and assumes the relationship between the independent variables after the logistic transformation and the dependent variables is linear. It is not constrained by the dimensions of the co-occurrence matrix. Therefore, it is possible to inject more features into the model to enhance the system's expressive power.

One of the main assumptions of algorithms mentioned earlier is that input features are independent. However, this ideal assumption generally does not hold true in practical applications. The lack of modelling possible relationships among features is the main bottleneck that limits the performance of previous models. Therefore, many researchers began to investigate methods to exploit the intrinsic relationships of features. A natural idea is using polynomials. Poly2 model [3] is an early attempt at feature conjunction modelling. It considers all pairwise combinations of features and assigns a weight to each combination. Although it is essentially a generalized linear model, the introduction of second-order polynomials captures the feature interaction information to a certain extent. However, the brute force combination sharply increases the time complexity and does not perform well on sparse data. Factorization machines (FMs) [35] replace the single weight coefficient in Poly2 with the inner product of latent weight factors. FMs' core idea is similar to matrix factorization — improving the generalization ability of a model by mapping features to the latent space. However, compared to matrix factorization, which only focuses on user and item features, FMs can learn the implicit

vector representation of all features. Fieldaware factorization machines (FFMs) [16] are improved variants of FMs. The basic assumption of FFMs is the learned feature vectors are not completely independent. Some vectors have a higher correlation with each other at a higher level than with others. Therefore, FFMs create a higher-level hierarchy called fields and group similar latent representations into the same field. The output of the recommender system is determined by single latent vectors as well as fields.

With the development of the deep learning field, more and more recommender systems improve the model performance by introducing neural networks at the architecture level. The combination of traditional recommendation algorithms' ideas and neural networks' expressive ability creates a new generation of more powerful recommender systems. AutoRec model [37] is an attempt to incorporate the autoencoders and the collaborative filtering algorithm. In AutoRec, user and item representations are learned by a single autoencoder. Experiments show that the introduction of autoencoders can enhance the model's ability to capture hidden patterns in data. Neural network-based collaborative filtering (NCF) [13] is also inspired by collaborative filtering but designs a more complex model architecture. The first part of NCF is multiple embedding layers that map user and item features into latent spaces. The second part is a generalized matrix factorization layer and a stack of multi-layer perceptrons (MLPs) to fuse the user embedding and the item embedding. There is a concatenation layer on the top of the model to combine the embeddings and generate prediction scores. Using neural networks to improve the feature conjunction modelling process is also an emerging research direction. FMs and FFMs generally can only explore the feature interactions through some variants of second-order polynomials. As a result, their expressivity is limited. Product-based neural networks (PNNs) [33] adopt the concept of fields in FFMs. It adds an embedding layer to encode features of fields and learns the possible interactions among fields by a product layer. Factorization-machine supported neural networks (FNN) [51] utilize the features learned by FMs to accelerate the convergence of the embedding layer. It further explores the relationships between features by several MLP layers. Neural Factorization Machines (NFMs) [12] completely discards the second-order part in FMs' core functions. Instead, it uses a bi-interaction pooling layer to simulate the interactions among features and enrich the model's expressivity.

The improvement of recommender systems benefits from the continuous progress of deep learning algorithms in different fields. Deep interest networks (DINs) [57] and deep interest evolution networks (DIENs) [56] are both deep learning recommenders proposed by Alibaba. Their main application scenario is placing advertisements on the e-commerce platform. DINs utilize the attention mechanism. The intuition behind DINs is that users

are more interested in advertisements which reflect their preferences. This means the design of the model should consider the high correlation between advertisement features and user features. The attention mechanism is a suitable technique to accomplish this task. DINs process user embeddings via a local attention mechanism module, which achieves the effect of personalized recommendations. The design of DIENs is inspired by the success of sequential neural networks. As is well known, the user's historical behavior data can be regarded as time series data. DIENs explore users' preferences by GRUs and capture shifts in users' behavior patterns by AUGRUs. Reinforcement learning has been a popular research area these years. DRNs [55] is a representative reinforcement learning-based recommender. In the DRN system, the agent is a deep Q-network and the environment is online users. The training process consists of the push stage and the feedback stage. Specifically, the agent recommends some items to the users at the push stage and then gathers users' feedback. The intensity of model updates is determined by users' feedback. One major advantage of DRNs is that they can quickly respond to dynamic changes in user preferences.

Before the era of deep learning, the performance of recommender systems was largely affected by the quality of manually selected features. Feature engineering is a time-consuming process and heavily depends on engineers' experience. Deep learning-based recommender systems, taking advantage of neural network's characteristics, tackle this challenge by adopting the end-to-end learning method. Deep Crossing model proposed by Microsoft [38] and Wide&Deep model proposed by Google [5] are both examples of end-to-end recommender systems applied in the industry. Deep learning also inspires the emergence of more efficient approaches to encode features. Item2vec [1] is a generalization of the Word2vec encoding method in the recommender system field. Given a sequence of historical data, it generates an item's latent representation by calculating the sum of item logarithmic probabilities. In addition to sequential data embedding learning, graph embedding learning is also a hot research topic. Most of online user-generated data is unstructured and can be represented as graphs. Therefore, informative graph embeddings are helpful for building recommender systems with promising applications. Commonly used graph embedding learning algorithms include DeepWalk [32] and Node2vec [10]. DeepWalk algorithm acquires graph embeddings via random walk on item graphs. Node2vec algorithm is an improvement of DeepWalk. It generates embedding sequences by using both breadth-first search and depth-first search, which reflect the homophily and structural equivalence of the graph respectively. EGES proposed by Alibaba [45] successfully incorporates graph embeddings into the recommender system. It performs a DeepWalk-style random walk on item graphs constructed from the user's historical behavior data. Besides, it enhances the basic graph embeddings with several

side information to alleviate the impact of the cold start problem.

2.2 Dialogue systems

Dialogue systems are agents that accept natural language inputs from users, transform input signals into some actions, and simulate human responses. Based on the application scenarios, there are two main kinds of dialogue systems: task-oriented and open-domain [30]. Task-oriented dialogue systems are designed for a specific purpose. In most cases, the goal of task-oriented agents is well-defined, for example, assisting online users to complete the reservation or search for related products. Open-domain dialogue systems, on the other hand, are expected to hold casual conversations with users covering a wide range of topics.

Early dialogue systems all belong to the task-oriented type [43]. They generate answers either based on hand-crafted rules or traditional machine learning models, whose state spaces are very limited. Although the restrictions on application scenarios and tasks alleviate the shortcomings of this type of model's insufficient expressivity, they can only interact with users in simple tasks and can not understand complex requests. The success of sequential neural networks promotes the development of end-to-end task-oriented dialogue systems. The reason is that generating responses based on user inputs is essentially a sequence-to-sequence task. Building a deep learning-based task-oriented dialogue system can be broken down into three consecutive steps: understanding language, learning policy, and generating responses. End-to-end dialogue systems proposed by [47] and [24] both follow this pipeline paradigm. [47] designs an embedding extractor composed of CNNs and RNNs to encode the natural language inputs, while [24] achieves the goal via LSTMs. They both create a database-augmented neural network to convert text embeddings into action vectors. Afterwards, LSTMs are deployed to translate the action signals into human-understandable sentences. The task-oriented dialogue system proposed by [54] tries to combine LSTMs and deep Q-networks from the reinforcement learning field. Deep Q-networks accept LSTMs' processed inputs and update weights at each turn.

In theory, deep neural networks are able to approximate any complex functions. They have the potential to expand the state space of dialogue systems, pushing the system performance beyond the limits of pre-defined templates or simple mathematical distributions. In this context, open-domain dialogue systems have attracted the attention of more and more researchers. Unlike task-oriented dialogue systems that strive to complete

the task in the minimum number of turns, open-domain dialogue systems can appropriately increase the number of turns to enhance the user's conversation experience. The open-domain dialogue system described in [39] is a representative early work. It uses the LSTM-based encoder-decoder framework to implement a pure generative dialogue system. Experiments show that it is able to generate better responses compared to models retrieving pre-defined responses from a database. The following work presented in [21] also adopts LSTMs as the backbone of the model. However, it not only considers word embeddings but also speaker embeddings when training the system. This feature augmentation improves the consistency of the system's responses during a multi-turn conversation. In most cases, the generative dialogue system outputs customized responses. However, it sometimes suffers from giving meaningless responses. Ensemble-GAN [50] leverages the adversarial learning technique to enable the generative system and the retrieval-based system to complement each other's strengths. Introducing external knowledge sources is also a direction to increase the intelligence of the dialogue system, especially in the scenario of keeping deep conversations with users. Researchers find that an encoder-decoder dialogue system equipped with knowledge data collected from the Internet generates more informative sentences than its counterparts even with larger parameter sizes [17].

It is worth mentioning that the advance of large language models blurs the distinction between the task-oriented system and the open-domain system. The design of dialogue systems based on large language models often takes both task-oriented and open-domain applications into account. Representative examples include BlenderBot 3 [40], ChatGPT based on GPT3.5 [2], and Llama 2 [42]. It is relatively easy for this kind of system to switch between small talk and conversations with clear purposes. This feature greatly increases their potential application scenarios.

2.3 Conversational recommender systems

The concept of conversational recommender systems (CRSs) benefits from the development of the recommender system field as well as the dialogue system field. Providing more personalized and high-quality recommended content is always the goal of recommender systems. Traditional recommender systems optimize the design process from a static perspective. Specifically, they continuously improve techniques to find more information contained in existing data. On the contrary, conversational recommender systems tackle the challenge from a dynamic perspective. The basic idea is that recommender systems can obtain more valuable information from the user, the largest information

source, through holding a conversation. There are two major advantages of discovering patterns from conversations directly. One is that systems' generated responses are more tailored to a specific user. The other is that recommender systems are able to quickly capture changes in user preferences.

Similar to the division of dialogue systems, conversational recommender systems can be divided into two categories based on the conversation pattern, namely attribute-aware CRSs and topic-guided CRSs [34]. Attribute-aware CRSs conduct conversations in a question-and-answer fashion. It proactively asks users about which product attributes users prefer, aiming at constructing user portraits as quickly as possible. However, this type of CRSs only accepts simple answers like yes or no, which tends to make conversations unnatural and limits the possibility of in-depth interactions. Topic-guided CRSs, in contrast, do not assume users have clear objectives at the beginning of the conversation. The strategy of topic-guided CRSs is collecting information about users from casual conversations and then guiding the conversation to the recommendation scenario. Interactions between topic-guided CRSs and users are more natural since systems can understand complex user requests. This project mainly focuses on the topic-guided CRSs. Therefore, the remainder of this section reviews related studies of topic-guided CRSs.

ReDial model [22] is a pioneering work in the conversational recommender system field. It is composed of two modules — a recommendation module and a conversation module. The conversation module is inspired by the hierarchical recurrent encoder-decoder framework. It maps input sentences into embeddings and then enhances the embeddings by sentiment-analyzing RNNs. Embeddings are delivered to the recommendation module for recommended item generation. After receiving the recommendation results, the conversation module generates complete answer sentences containing recommended items as the system's outputs. The recommendation part is implemented as an autoencoder. Its function is selecting items that best match user preferences based on embeddings provided by the conversation module. The operations of these two modules are complementary to each other. Most of the subsequent conversational recommender systems follow the paradigm of a recommendation module plus a conversation module.

There are two main directions for improving the overall performance of CRSs. One is using external knowledge to reduce uncertainty during the training. The other is modifying the model architecture to increase the information utilization rate. For topic-guided CRSs, the main external information sources include knowledge graphs and relevant reviews. KBRD [4] is an early attempt to introduce knowledge graphs into the model. Its assumption is that extra information about entities appearing in utterances is helpful

for both the recommendation task and the conversation task. KGSF [59] carries forward the idea of enriching word embeddings by leveraging knowledge graphs. It not only uses encyclopedic knowledge graphs but also maximizes the utilization of common-sense knowledge by adding semantic network-based knowledge graphs. Many of the following works incorporate at least one type of knowledge graphs into the system to enhance embeddings, such as UniCRS [46], CR-Walker [28], CRFR [58], etc. User reviews are also informative external data. Treating all user reviews for a specific item as a whole helps the system find users' average opinion about the item. Meanwhile, discovering differences between users through review helps the system provide personalized recommendations. RevCore [27] is a typical model that collects related reviews from the Internet, creates a database, and augments word embeddings with user reviews. There are also hybrid models that combine both knowledge graphs and user reviews, for example, C^2-CRS model [61].

Improving model architecture is critical to the development of conversational recommender systems. Many existing studies focus on embedding fusion mechanisms. As mentioned before, external knowledge has been widely applied in CRSs. Therefore, it is important to better integrate the external knowledge with conversation data. KBRD model extracts subgraphs of the original encyclopedic knowledge graph by performing the entity linking operation to associate entities that appear in conversations with nodes in the knowledge graph [4]. It encodes both the node and edge information via R-GCNs, since the relationship between two entities may offer some insight into user behaviors. Graph embeddings produced by R-GCNs are processed as inputs by recommendation and conversation modules. There is a switching network at the top of the model to merge knowledge from both of the two modules and generate responses. KGSF model handles encyclopedic knowledge graph encoding in a similar way. As for the semantic network-based knowledge graph, the semantic relationships among words tend to contain more noise than helpful information for the recommendation task. Thus, KGSF links words in conversations with the semantic network-based knowledge graph and chooses GCNs to encode node information only. Besides, KGSF designs a mutual information maximization (MIM) function to fuse embeddings from two types of knowledge graphs before sending embeddings to the recommender system and dialogue system. The MIM mechanism effectively solves the alignment problem between different latent spaces [59]. CR-Walker [28] and CRFR [58] model explore more complicated patterns between graph embeddings and user embeddings through graph reasoning. CR-Walker creates a walker cell to construct tree-structured reasoning paths on the concatenation of graph embeddings and context embeddings. It defines a set of reasoning rules covering two-hop neighbors to trade-off between efficiency and accuracy. CRFR introduces actor-critic

networks to infer connections among higher-order hop neighbors. Flexibility brought by the policy reasoning strengthens the model's robustness even when knowledge graphs are incomplete. Models utilizing review information usually perform lookup operations to achieve the embedding alignment. For instance, RevCore aligns review, context, and entity embeddings through a lookup table [27]. C^2-CRS model further investigates the embedding fusion from the level of information granularity. It breaks down information from various sources into different granularities and deploys contrastive learning to seamlessly fuse embeddings at each granularity [61].

There are other perspectives on improving the model architecture. UCCR [23] proposes to learn user preferences from the current conversation as well as the historical conversations. Given the historical data, it extracts information from three dimensions: semantics, entities, and items. Multi-view embeddings learned from history are complementary to embeddings learned from the current conversation, which has the benefit of static data and dynamic interactions. Both UPCR [34] and VRICR model [52] use the variational inference. UPCR discovers user preferences over short-term and long-term time intervals. VRICR addresses the problem of reducing the impact of incomplete knowledge graphs by performing variational reasoning. Some studies explore the possible application of pre-trained large language models in the CRSs field. MESE [49] leverages the power of the pre-trained language model to fuse meta-data into the context embeddings. UniCRS [46] tries to unify the training process of the recommendation and conversation module with the help of DialoGPT.

This project focuses on the reproducibility of four representative CRSs: KBRD, KGSF, UniCRS, and C^2-CRS.

Chapter 3

Data collection

3.1 Datasets

This project mainly uses three conversational recommendation datasets: ReDial [22], TG-ReDial [60], and INSPIRED [11]. ReDial and INSPIRED are English datasets, while TG-ReDial is a Chinese dataset. Details of datasets are provided below.

3.1.1 ReDial

ReDial is the first public large-scale dataset specifically created for real-world conversational recommendation tasks, filling the gap in traditional recommendation datasets that lack interaction data with users. It mainly focuses on the movie recommendation scenario. The authors of ReDial recruited several native English speakers from a crowdsourcing platform called Amazon Mechanical Turk (AMT). They required a pair of workers to have a conversation about movie recommendations, where one person played the role of recommender and the other played a movie seeker. The length of conversations was limited to around ten rounds, and workers were expected to mention at least four movies during the conversation. Since workers discuss movies in a natural way in these conversations, they can adjust the direction of the dialogue according to the other person's responses. Therefore, the ReDial dataset is very close to the recommendation needs of users in daily life. The dataset is published in JSONL format. Each record includes the dialogue ID, the time offset, text, mentioned movies, the message sender's ID, and emotion labels. Table 3.1 shows the statistics of ReDial.

Table 3.1: Statistics of ReDial, TG-ReDial, and INSPIRED datasets.

		Total	
	ReDial	**TG-ReDial**	**INSPIRED**
# Users	956	1,482	1,594
# Conversations	10,006	10,000	1,001
# Utterances	182,150	129,392	35,811
# Movies	51,699	33,834	5,382
# Unique tokens	26,736	46,973	18,316
		Average	
	ReDial	**TG-ReDial**	**INSPIRED**
# Turns per conversation	9.58	6.97	10.73
# Tokens per utterance	14.74	19.00	17.96

3.1.2 TG-ReDial

Similar to the ReDial dataset, the TG-ReDial dataset is designed to address the recommendation task within conversation scenarios. However, TG-ReDial introduces a topic guidance mechanism based on daily conversations. It assumes that conversation contents in the conversational recommendation task can be summarized as a series of topics. Some branches of topics keep the interaction as chit-chat, while others finally lead the conversation to a successful movie recommendation. TG-ReDial mainly collects raw data from the Douban website, a popular Chinese social networking site for book, movie, and music reviews. First, the creators of TG-ReDial gather lots of user profiles and users' movie-watching history. Afterward, they simulate the evolution of topics during conversations and associate user profiles as well as movies with topics. Then, they invite some workers to manually complete conversations according to given topic threads. The dataset creator finally provides the data in PKL format, where it includes basic information such as conversation ID, text, and movies mentioned. The dataset also annotates the recommended items, sentiment, and intent labels. Compared to the ReDial dataset, the conversation content of TG-ReDial is more diverse, covering multiple fields such as movies, music, books, and travel. Table 3.1 summarizes the information of TG-ReDial in detail.

3.1.3 INSPIRED

Inspired is a conversational recommendation dataset that emphasizes real-time interaction and sentiment analysis. It focuses on social strategies that increase the recommendation success rate during conversations. According to the theory of human behavior, the dataset creator divides the recommendation strategies into two categories: sociable strategies and preference elicitation strategies. Sociable strategies aim to quickly build a trust relationship with users at the start of a session, enhancing their confidence in the recommendation system. There are eight types of sociable strategies: personal opinion, personal experience, similarity, encouragement, offering help, preference confirmation, self-modeling, and credibility. Preference elicitation strategies are conversation strategies used to better identify user preferences when the conversation reaches the recommendation stage. These strategies include experience inquiry and opinion inquiry. For data collection, records in INSPIRED are created in a similar way as ReDial. Specifically, two workers simulate the conversation between a recommender and a movie seeker. However, authors add social strategy tags to each conversation based on social science research. The INSPIRED dataset is provided in TSV format, which contains the dialogue ID, movie mentioned, text, and social strategy labels at varying levels of granularity. Table 3.1 shows details of INSPIRED.

3.2 Knowledge graphs

As mentioned in Section 2.3, CRSs mainly incorporate two types of knowledge graphs: encyclopedic knowledge graphs and semantic network-based knowledge graphs. These two types of knowledge graphs are categorized based on their structure and application domains [15]. Encyclopedic knowledge graphs focus mainly on real-world entities and their relationships. They are generally used for knowledge management and retrieval. These knowledge graphs typically extract information from online encyclopedic resources, with nodes representing entities such as people, time, and events. DBpedia [19] and CN-DBpedia [48] are two of the most commonly used encyclopedic knowledge graphs for enriching entity-related knowledge. In contrast, semantic network-based knowledge graphs emphasize the semantic relationships between words and are primarily applied in natural language processing tasks. ConceptNet [41] and HowNet [6] are among the most popular semantic network-based knowledge graphs. This section introduces these four knowledge graphs in detail.

3.2.1 DBpedia

DBpedia is a multi-language knowledge graph database constructed from Wikipedia articles. We mainly use English DBpedia ontology in this project. The DBpedia ontology is the backbone of the DBpedia project. It collects raw data from Wikipedia and then categorizes the entities in the data. The DBpedia ontology is implemented as a hierarchical structure with a maximum depth of five allowed. Mapping rules between the Wikipedia source and the DBpedia ontology are maintained by the DBpedia crowdsourcing community. The DBpedia foundation renews the snapshot of DBpedia ontology every day. In order to fairly compare the performance between different models, we adopt the version of the DBpedia ontology used in KBRD's source code[1]. This DBpedia ontology includes 4,828,418 entities, which are grouped into 768 classes. Besides, there are 18,746,176 relations among entities. According to mapping rules, 3,000 properties are designed to describe entities. Figure 3.1 shows the percentage and counts of entities in several major categories.

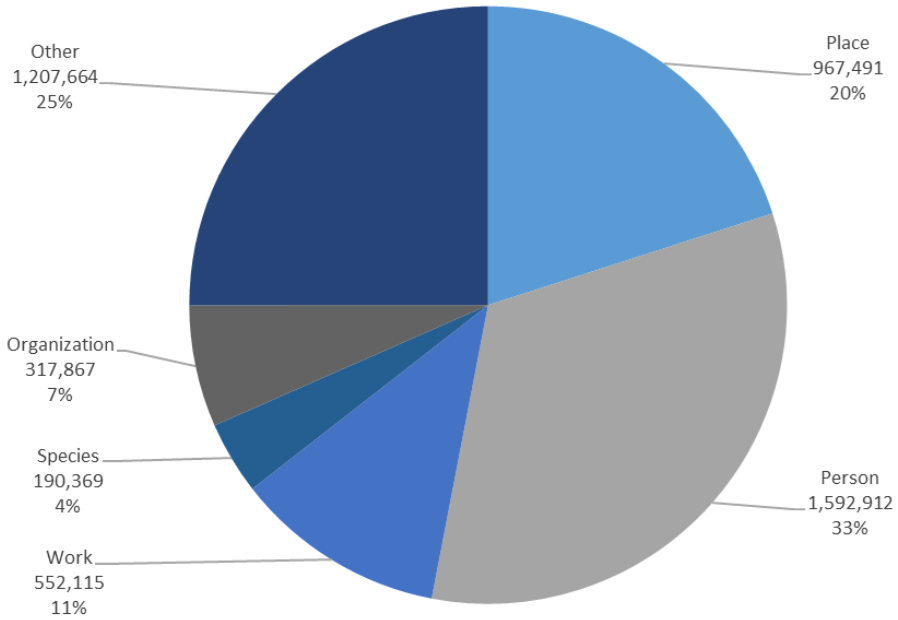

Figure 3.1: Distribution of entities over major categories in DBpedia ontology.

3.2.2 CN-DBpedia

CN-DBpedia has been the largest public Chinese knowledge graph database so far. It is an upgraded version of Fudan GDM Chinese knowledge graphs. Its data sources include

[1]DBpedia snapshot: https://drive.google.com/file/d/1WqRoQAxH_kdoJpbYVsFF0EN4ZJxiiDB2/view

three popular Chinese knowledge encyclopedias: Chinese Wikipedia, Hudong Baike, and Baidu Baike. The knowledge base is established by following a pipeline method. To be specific, the raw data is expected to flow through extraction, normalization, enrichment, and correction modules before entering the database. The processed record is presented as a tab-separated triple, i.e. (entity's name, attribute's name, attribute's value). There are 10,341,196 entities and 88,454,264 relations in CN-DBpedia. Authors of CN-DBpedia offer two ways to access records: sending requests via CN-DBpedia APIs or using the dump file. The CN-DBpedia server requires the API key if a client sends a large number of requests in a short period of time. Therefore, we choose the dump file[2] as the data source for this project. Figure 3.2 presents the top fifteen most popular classes in CN-DBpedia.

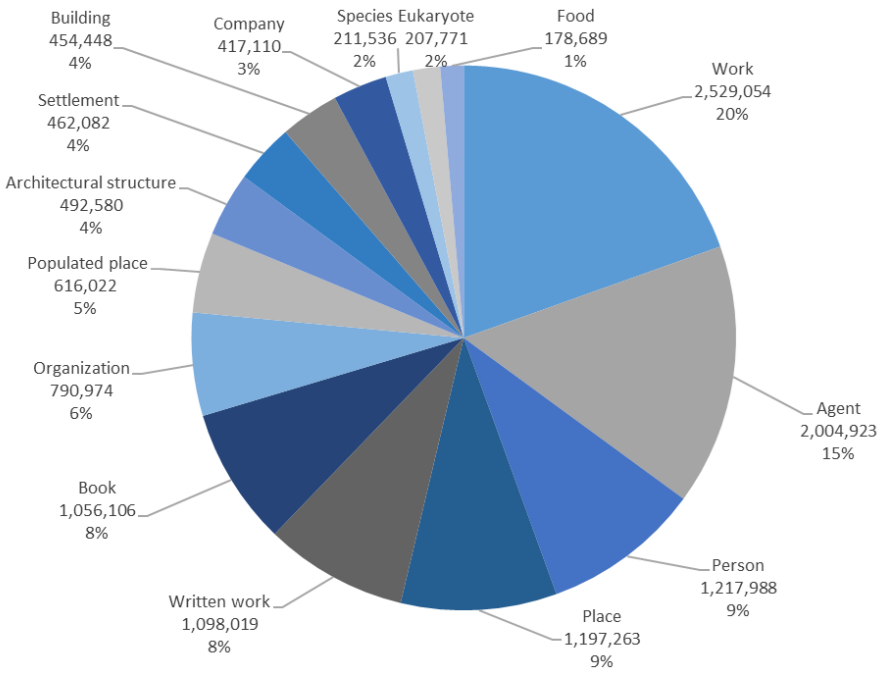

Figure 3.2: Distribution of entities over major categories in CN-DBpedia.

3.2.3 ConceptNet

ConceptNet is a semantic network-based knowledge graph organized according to semantics. The main data sources of ConceptNet include Open Mind Common Sense, Wiktionary, WordNet, JMDict, OpenCyc, and a subset of DBpedia. Users can easily access the ConceptNet database by sending REST API calls[3]. The result returned by the server is a JSON string containing meta information and edges connected to the

[2]CN-DBpedia dump file: http://kw.fudan.edu.cn/cndbpedia/download/
[3]ConceptNet APIs: https://github.com/commonsense/conceptnet5/wiki/API

query word. It supports queries in 83 languages, but we only use English keywords in our work. ConceptNet has approximately 8 million nodes and 21 million edges. For English words, there are over 1,500,000 nodes.

3.2.4 HowNet

HowNet is a Chinese semantic network-based knowledge graph that focuses on exploring relationships between Chinese words. Its main data source is the Chinese language knowledge database of the same name. The creators of HowNet define 94 semantic relations in total. There are 100,385 Chinese words and 192,191 edges in HowNet, covering 7,182 unique Chinese characters. The OpenHowNet project provides a complete set of Python APIs[4] to query HowNet data, which is the way we choose to access HowNet.

3.2.5 Comparison of knowledge graphs

This section compares data sources, data structures, and construction methods of the four knowledge graphs to provide a more comprehensive overview.

Data sources. The data of DBpedia and CN-DBpedia mainly come from the entries of the target language in online encyclopedias. ConceptNet also draws much of its data from online encyclopedias but integrates additional sources like online dictionaries. HowNet primarily leverages existing linguistic lexicons.

Data structures. DBpedia and CN-DBpedia are both encyclopedic knowledge graphs. In order to highlight real-world entities and the relationships between them, both of them use entity-relation-entity triples to store the graph data. ConceptNet and HowNet, as representatives of semantic network-based knowledge graphs, focus more on the semantic relations among words. Each record in these knowledge graphs typically corresponds to a single word and includes information such as the word's definition, synonyms, antonyms, usage examples, and conceptually related words.

Construction methods. DBpedia and CN-DBpedia both crawl the latest encyclopedia data automatically and update their databases regularly, which enables them to release new versions faster. ConceptNet also uses automated updates but introduces crowd-sourcing to improve the accuracy of semantic connections. However, most of the records in HowNet are constructed through manual annotation, since HowNet relies heavily on

[4]OpenHowNet APIs: https://github.com/thunlp/OpenHowNet

manually predefined semantic rules.

Chapter 4

Methodology

In this chapter, we describe the method we used to control the amount of external knowledge in CRSs. Figure 4.1 shows an overview of our approach. Section 4.1 explains the process of creating the seed set and building n-hop subgraphs. Section 4.2 introduces how we simulate the scenario of incomplete knowledge graphs in detail.

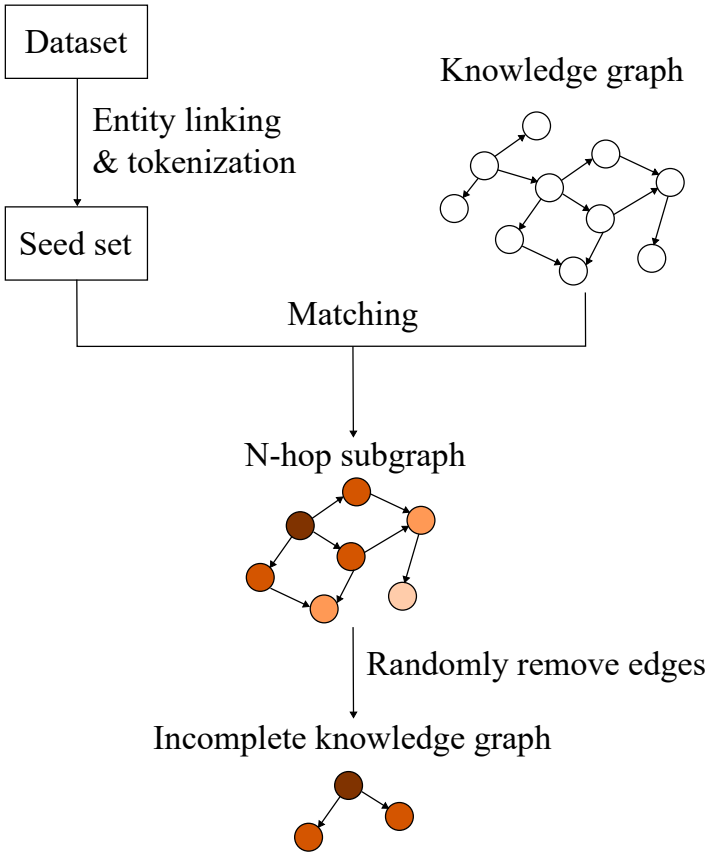

Figure 4.1: The overview of our method to control the amount of external knowledge.

4.1 N-hop subgraph construction

According to RQ1, our first goal is to quantify the information brought by the external knowledge graph. We notice that many knowledge graphs are directed acyclic graphs. Thus, a natural idea is to use some kind of graph traversal algorithm to extract subgraphs related to entities appearing in the dataset. Inspired by RippleNet [44], our strategy is finding n-hop neighbors of the seed in the knowledge graph by the BFS algorithm. RippleNet defines the seed set as the union of user click history. It assumes that the influence of user preferences will first be transmitted to adjacent nodes on the knowledge graph. Then just like ripples, user preferences are propagated outward layer by layer in the knowledge graph. We define the seed set as the union of entities or selected words appearing in the conversation history of the dataset in our project. Considering that the impact of user preferences will gradually weaken as the layers of BFS increase, we adopt one-hop, two-hop, and three-hop subgraphs to represent different amounts of external knowledge. To sum up, the n-hop subgraph construction is a two-step process: the creation of the seed set and the extraction of subgraphs.

Benchmark datasets in the CRSs field usually contain historical conversation records between users and the system. As mentioned in Section 3.2, we have encyclopedic knowledge graphs and semantic network-based knowledge graphs, so we need to prepare two types of seed sets accordingly. For the seed set of encyclopedic knowledge graphs, we can use entity linking techniques to extract entities in the conversation text. There are already many mature entity linking tools on the market, such as DBpedia Spotlight for English and ChineseNLP for Chinese. In experiments, we adopt the entity linking results of the ReDial dataset given in the KBRD code implementation[1] and the entity linking results of the TG-ReDial and INSPIRED datasets given in the CRSLab code implementation[2] to be consistent with most existing works. For the seed set of semantic network-based knowledge graphs, we first tokenize sentences in the CRS dataset and then filter out all stop words. Note that query results of ConceptNet usually contain related nodes in other languages. Thus, we only retain nodes with the language label 'en'. In addition, we set a threshold of 10 to filter out nodes with few connections.

After obtaining the seed set, we query the knowledge database to construct the entity set $\mathcal{N}_0$, which includes entities that appear in both the dataset and the knowledge graph. Next, we find the n-hop neighborhoods of node v for $\forall v \in \mathcal{N}_0$ through the BFS algorithm.

[1]KBRD: https://github.com/THUDM/KBRD
[2]CRSLab: https://github.com/RUCAIBox/CRSLab/tree/main

It can be expressed as:

$$\mathcal{N}_k = \{v_2 \mid (v_1, r, v_2) \in \mathcal{G} \wedge v_1 \in \mathcal{N}_{k-1}\}, \ k = 1, 2, 3, \tag{4.1}$$

where (v_1, r, v_2) represents a directed edge from node v_1 to node v_2 with the attribute r in the knowledge graph $\mathcal{G}$. Given $\mathcal{N}_k$, we can define the n-hop subgraph $\mathcal{S}_k$ as:

$$\mathcal{S}_k = \{(v_1, r, v_2) \mid v_1 \in \mathcal{N}_{k-1} \wedge v_2 \in \mathcal{N}_k\} \cup \mathcal{S}_{k-1}, \ k = 1, 2, 3, \tag{4.2}$$

where $\mathcal{S}_0 = \emptyset$. In the code implementation, we store subgraphs in the form of edge lists to satisfy the input format requirements of CRSs. Table 4.1 summarizes the statistics of n-hop subgraphs we extract from DBpedia, CN-DBpedia, ConceptNet, and HowNet.

Table 4.1: Statistics of n-hop subgraphs extracted from four knowledge graphs.

		DBpedia	CN-DBpedia	ConceptNet	HowNet
One-hop	# nodes	36,226	81,173	28,208	21,291
	# edges	72,330	163,261	47,023	44,376
	# relations	79	55	40	87
Two-hop	# nodes	87,531	113,581	69,198	21,338
	# edges	165,743	258,700	168,669	74,672
	# relations	214	55	44	89
Three-hop	# nodes	139,981	123,907	115,392	23,395
	# edges	297,052	299,364	347,941	105,033
	# relations	410	55	44	89

Most of CRSs encode encyclopedic knowledge graphs with R-GCNs, while encoding semantic network-based knowledge graphs with GCNs. Therefore, the model training process usually only uses edge attributes of DBpedia and CN-DBpedia.

4.2 Incomplete knowledge graph construction

In real-world scenarios, the complete knowledge graph is not always accessible due to limited data resources, insufficient database permissions, etc. Therefore, it is important to select an appropriate model according to the robustness. In order to examine the model's robustness and answer the RQ2, we simulate the scenario of incomplete knowledge graphs by randomly removing 10%, 30%, and 50% edges in the one-hop subgraph $\mathcal{S}_1$ of DBpedia, CN-DBpedia, ConceptNet, and HowNet. We choose to remove edges instead of nodes because one record stored in the knowledge database corresponds to

one edge in the knowledge graph in most cases. That means the incompleteness of the knowledge graph is most likely caused by missing edges. The statistics of incomplete knowledge graphs are shown in Table 4.2.

Table 4.2: Statistics of incomplete knowledge graphs.

		DBpedia	CN-DBpedia	ConceptNet	HowNet
10% drop rate	# nodes	27,135	58,836	20,404	14,427
	# edges	65,097	146,934	42,320	39,938
	# relations	71	55	40	86
30% drop rate	# nodes	24,705	53,319	18,391	12,683
	# edges	50,631	114,282	32,916	31,063
	# relations	69	55	40	84
50% drop rate	# nodes	21,545	46,017	15,744	10,581
	# edges	36,165	81,630	23,511	22,188
	# relations	61	55	40	81

Chapter 5

Experiments and results

In this chapter, we present the whole process and the results of our experiments. Section 5.1 describes our experimental settings. Section 5.2 lists all metrics we use in the project to evaluate models' performance on recommendation and conversation tasks. We comprehensively consider the metrics used in existing work on CRSs in recent years when selecting the metrics for our experiments. Section 5.3 summarizes our experimental results on n-hop knowledge graphs and incomplete knowledge graphs. We also show some sample responses generated by CRSs when the amount of external knowledge varies in the case study.

5.1 Experimental set-up

We deploy all experiments on a cloud server with an NVIDIA GeForce RTX 3090 GPU. The main third-party libraries used in experiments are PyTorch, PyTorch Geometric, and Transformers. The original code of KBRD and KGSF only provides the implementation on the ReDial dataset. Therefore, we fork CRSLab's version[1] of KBRD and KGSF, which has integrated three datasets. For UniCRS and C^2-CRS, we develop our implementation based on their original code[2]. Our modified model implementation fixes the bugs in the original version, supports knowledge graphs of different sizes, and adds more comprehensive evaluation metrics. We also add a series of shell scripts and the model checkpoint saving/loading feature to automate the process of model deployment. Our code implementation is available on GitHub: https://github.com/ShupeiLi/CRSLab

[1] CRSLab: https://github.com/RUCAIBox/CRSLab/tree/main

[2] UniCRS: https://github.com/RUCAIBox/UniCRS, C^2-CRS: https://github.com/Zyh716/WSDM2022-C2CRS/tree/main

(KBRD and KGSF), https://github.com/ShupeiLi/UniCRS (UniCRS), https://github.com/ShupeiLi/WSDM2022-C2CRS (C^2-CRS). For most of the hyperparameter settings, we adopt the values recommended by the authors of the original paper. However, we fine-tune the model's learning rate and training epochs to ensure the performance of our implementation is consistent with that reported in the original paper. Table 5.1 summarizes our hyperparameter settings. Besides, we set the random seed to 42 when randomly removing edges of knowledge graphs in order to ensure reproducibility.

Table 5.1: Hyperparameter settings used in our experiments.

Model	**Dataset**	**Task**	**Parameter**	**Value**
KBRD	ReDial	recommendation	learning rate	3×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
		conversation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
	TG-ReDial	recommendation	learning rate	3×10^{-3}
			epochs	100. Enable the early stopping with patience 3.
		conversation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 3.
	INSPIRED	recommendation	learning rate	3×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
		conversation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
KGSF	ReDial	pre-training	learning rate	1×10^{-3}
			epochs	3
		recommendation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
		conversation	learning rate	1×10^{-3}
			epochs	100. Enable the early

Model	Dataset	Task	Parameter	Value
				stopping with patience 5.
	TG-ReDial	pre-training	learning rate	1×10^{-3}
			epochs	50
		recommendation	learning rate	1×10^{-3}
			epochs	20. Enable the early stopping with patience 3.
		conversation	learning rate	1×10^{-3}
			epochs	10
	INSPIRED	pre-training	learning rate	1×10^{-3}
			epochs	3
		recommendation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
		conversation	learning rate	1×10^{-3}
			epochs	100. Enable the early stopping with patience 5.
UniCRS	ReDial	pre-training	learning rate	5×10^{-4}
			epochs	5
		recommendation	learning rate	1×10^{-4}
			epochs	5
		conversation	learning rate	1×10^{-4}
			epochs	10
	INSPIRED	pre-training	learning rate	6×10^{-4}
			epochs	5
		recommendation	learning rate	1×10^{-4}
			epochs	5
		conversation	learning rate	1×10^{-4}
			epochs	10
C^2-CRS	ReDial	pre-training	learning rate	1×10^{-3}
			epochs	25
		recommendation	learning rate	1×10^{-3}
			epochs	50. Enable the early stopping with patience 3.
		conversation	learning rate	1×10^{-3}

Model	Dataset	Task	Parameter	Value
	TG-ReDial	pre-training	epochs	30
			learning rate	1×10^{-3}
		recommendation	epochs	25
			learning rate	1×10^{-3}
			epochs	50. Enable the early stopping with patience 3.
		conversation	learning rate	1×10^{-3}
			epochs	22

5.2 Evaluation metrics

5.2.1 Recommendation task

For the recommendation task, we evaluate the performance of models by Recall@k (k = 1, 10, 50), MRR@k (k = 10, 50), and NDCG@k (k = 10, 50). Positive samples are the top-k items in the ground-truth label set.

Recall@k. Recall@k reflects the percentage of the number of correctly predicted relevant items to the number of truly relevant items.

$$\text{Recall@}k = \frac{\#\text{ correctly predicted relevant items}}{\#\text{ truly relevant items}},\ k = 1, 10, 50. \tag{5.1}$$

MRR@k. MRR is an abbreviation of mean reciprocal rank. It uses rankings rather than counts to evaluate the quality of recommendation results. MRR@k is calculated as follows:

$$\text{MRR@}k = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{rank}_i},\ k = 10, 50. \tag{5.2}$$

where N represents the number of queries and rank_i denotes the rank of the first truly relevant item for the query i. If $\text{rank}_i > k$, we set the value of the reciprocal rank term to 0.

NDCG@k. NDCG is also known as normalized discounted cumulative gain. Its main idea is that correctly predicted relevant items with higher ranks contribute more to the quality of results. The final score is the ratio of the actual discounted cumulative gain

to the ideal discounted cumulative gain, which is between 0 and 1.

$$\text{NDCG@}k = \frac{\text{DCG}}{\text{IDCG}} = \frac{\sum_{i=1}^{n} \frac{\text{score}_i}{\log(i+1)}}{\sum_{i=1}^{|R|} \frac{1}{\log(i+1)}}, \ k = 10, 50. \tag{5.3}$$

$$\text{score}_i = \begin{cases} 1, \ \text{if item } i \text{ is relevant and its rank is less than } k, \\ 0, \ \text{otherwise.} \end{cases} \tag{5.4}$$

where $|R|$ is the number of relevant items whose ranks are less than k.

5.2.2 Conversation task

We use automatic metrics to evaluate the performance of models on the conversation task. After comprehensively considering the experimental reports of existing work, we select the following intrinsic metrics for the generated utterances: BLEU@k ($k = 1, 2, 3, 4$), ROUGE@k ($k = 1, 2, L$), intra-distinct@k ($k = 1, 2, 3, 4$), and inter-distinct@k ($k = 1, 2, 3, 4$). It is important to note that these metrics are used to compare the generated utterances with the real utterances in the dataset.

BLEU@k. BLEU [31] was first proposed for evaluating the quality of machine translation systems' results, and has since been widely applied to the evaluation of dialogue systems. For the generated sentence evaluation, BLEU's basic assumption is that the model performance is positively related to the number of n-gram overlaps between the generated responses and the ground truth responses. We denote the generated sentences as $\hat{S} = (\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_m)$ and the ground truth references as $S = (S_1, S_2, \ldots, S_m)$, where $S_i = (s_{i,1}, s_{i,2}, \ldots, s_{i,n_i}), \ i = 1, 2, \ldots, m$. The first step of BLEU@k is to calculate the modified precision score p_k:

$$p_k = \frac{\sum_{i=1}^{m} \sum_{y \in G_k(\hat{s}_i)} \min\left(C(y, \hat{s}_i), \max_{s \in S_i} C(y, s)\right)}{\sum_{i=1}^{m} \sum_{y \in G_k(\hat{s}_i)} C(y, \hat{s}_i)} \tag{5.5}$$

where $G_k(a)$ is a function that generates the k-gram set of sentence a and $C(a, b)$ counts the number of occurrences of subsequence a in string b. And after that, we need to compute the value of the brevity penalty term:

$$bp = \begin{cases} 1, \ \text{if } c > r, \\ e^{1-\frac{r}{c}}, \ \text{otherwise.} \end{cases} \tag{5.6}$$

c and r in Equation 5.6 represent the length of generated responses and the effective

length of the references, respectively. That is,

$$c = \sum_{i=1}^{m} |\hat{s}_i|, \tag{5.7}$$

$$r = \sum_{i=1}^{m} \min_{s \in S_i} ||s| - |\hat{s}_i|| \tag{5.8}$$

where $|a|$ denotes the length of sentence a. BLEU@k is a weighted geometric mean of the modified precision scores $p_i, i = 1, 2, \ldots, k$. Given a weight vector $w = (w_1, w_2, \ldots, w_k)$, we can compute the BLEU@k as follows:

$$\text{BLEU@}k = bp \cdot \exp\left(\sum_{i=1}^{k} w_i \log p_i\right), \ k = 1, 2, 3, 4 \tag{5.9}$$

We compute the sentence-level BLEU scores with the help of NLTK package[3] in our experiments. It is interesting that we find the selection of the weighted vector is inconsistent in the existing work. Specifically, some papers use the one-hot vector while others use the vector with uniform weights. We adopt the latter option since it is more commonly used.

ROUGE@k. ROUGE [25] is a set of metrics proposed to evaluate the quality of text summarization. We use ROUGE@$\{1, 2, L\}$ in our experiments. ROUGE@k is essentially the n-gram recall. We follow the notation specified in the BLEU calculation. The mathematical expression for ROUGE@k can be written as:

$$\text{ROUGE@}k = \frac{\sum_{i=1}^{m} \sum_{s \in S_i} \text{Count}(G_k(s) \cap G_k(\hat{s}_i))}{\sum_{i=1}^{m} \sum_{s \in S_i} \text{Count}(G_k(s))}, \ k = 1, 2. \tag{5.10}$$

where the function $\text{Count}(a)$ returns the number of elements in set a. As for ROUGE@L, it uses the longest common subsequence instead of the n-gram tokens. That is,

$$\text{ROUGE@}L = \frac{1}{m} \sum_{i=1}^{m} \sum_{s \in S_i} \frac{\text{LCS}(s, \hat{s}_i)}{\text{Count}(G_1(s))} \tag{5.11}$$

where the function $\text{LCS}(a, b)$ computes the longest common subsequence between string a and string b. In our project, we calculate the ROUGE scores with the help of the py-rouge library[4].

Distinct@k. Distinct proposed in [20] is a metric aiming at measuring the diversity of

[3]NLTK bleu score: https://www.nltk.org/_modules/nltk/translate/bleu_score.html
[4]py-rouge: https://github.com/Diego999/py-rouge

generated responses. It reflects the ratio of the number of unique n-gram tokens to the total number of n-gram tokens. We refer to dialogue metrics in the ParlAI framework proposed by Facebook [29] and compute the intra-distinct@k as well as inter-distinct@k scores. Intra-distinct@k is the sample-level distinct score, while inter-distinct@k is the global-level one. Given the generated sentences $S = (\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_m)$, we can compute distinct scores as follows:

$$\text{Intra-Distinct@}k = \frac{1}{m}\sum_{i=1}^{m}\frac{\text{Count}_{\text{unique}}(G_k(\hat{s}_i))}{\text{Count}(G_k(\hat{s}_i))}, \tag{5.12}$$

$$\text{Inter-Distinct@}k = \frac{\text{Count}_{\text{unique}}(\bigcup_{i=1}^{m} G_k(\hat{s}_i))}{\text{Count}(\bigcup_{i=1}^{m} G_k(\hat{s}_i))}, \ k = 1, 2, 3, 4. \tag{5.13}$$

where $\text{Count}_{\text{unique}}(a)$ returns the number of unique elements in set a. Most existing work only reports the inter-distinct scores. We notice that many of them use the number of samples instead of the number of tokens as the denominator in the code implementation, which leads to an inter-distinct score greater than 1. We think their calculation method is incorrect because a ratio can only be a value between 0 and 1. In our experiments, we use Equation 5.12 and Equation 5.13 to compute distinct scores, thus avoiding the above-mentioned bug.

5.3 Results

5.3.1 Impact of n-hop knowledge graphs

We first investigate the performance of four models on the recommendation task when the amount of external knowledge changes. Table 5.2 - 5.7 summarize the experimental results. The '1-hop', '2-hop', and '3-hop' in the header represent using one-hop, two-hop, and three-hop knowledge subgraphs respectively.

From Table 5.2, we can see that the increase in external knowledge from DBpedia generally enhances the performance of KBRD and C^2-CRS across most recommendation metrics on the ReDial dataset. On the contrary, adding more external knowledge impairs the performance of UniCRS on the recommendation task. For the KGSF model, the two-hop subgraph of DBpedia is the most suitable setting for external knowledge integration. However, these observations for DBpedia settings on the ReDial dataset do not apply to the INSPIRED dataset. According to Table 5.6, the performance of KBRD and KGSF decreases as the size of the DBpedia subgraph increases, while UniCRS achieves

Table 5.2: Recommendation task: experimental results on the ReDial dataset with n-hop DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
Recall@1	0.035	0.035	**0.039**	0.031	**0.035**	0.032	0.043	**0.047**	0.017	**0.051**	0.050	0.048
Recall@10	0.178	**0.187**	0.185	0.172	**0.178**	0.177	**0.212**	0.202	0.052	0.211	0.228	**0.231**
Recall@50	0.330	0.342	**0.345**	0.352	0.364	**0.367**	**0.410**	0.409	0.085	0.388	0.391	**0.408**
MRR@10	0.071	0.072	**0.074**	0.066	**0.071**	0.068	**0.087**	**0.087**	0.027	0.092	0.095	**0.096**
MRR@50	0.078	0.080	**0.082**	0.075	**0.080**	0.078	0.096	**0.097**	0.028	0.101	0.103	**0.104**
NDCG@10	0.095	0.099	**0.100**	0.091	**0.096**	0.094	**0.116**	0.114	0.033	0.120	0.126	**0.127**
NDCG@50	0.129	0.133	**0.135**	0.130	**0.137**	0.136	**0.160**	**0.160**	0.040	0.159	0.162	**0.166**

Table 5.3: Recommendation task: experimental results on the ReDial dataset with n-hop ConceptNet knowledge graphs.

	KGSF			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
Recall@1	0.033	0.033	**0.034**	0.052	**0.056**	0.052
Recall@10	**0.180**	0.173	0.178	0.230	**0.236**	0.224
Recall@50	**0.365**	0.359	0.364	0.406	**0.415**	0.405
MRR@10	**0.069**	0.068	**0.069**	0.099	**0.103**	0.098
MRR@50	**0.078**	0.077	0.077	0.107	**0.111**	0.106
NDCG@10	**0.095**	0.093	0.094	0.130	**0.134**	0.127
NDCG@50	**0.136**	0.134	0.135	0.169	**0.174**	0.168

the best performance with two-hop DBpedia subgraphs. Table 5.4 illustrates how CN-DBpedia, another encyclopedic knowledge graph, impacts the model performance on the TG-ReDial dataset. The KBRD system does not benefit from the increased size of the CN-DBpedia subgraph. However, the performance of KGSF and C^2-CRS shows a slight improvement in the three-hop subgraph setting after a decline in the two-hop subgraph setting. Next, we analyze the role of n-hop common sense knowledge graphs in model performance on the recommendation task. Table 5.3 shows that the performance of KGSF does not differ much with the use of one-hop and three-hop ConceptNet subgraphs, while C^2-CRS performs best when using the two-hop subgraphs. Table 5.7 reflects that KGSF's performance on the INSPIRED dataset benefits from introducing more information from ConceptNet. In particular, its performance on MRR@10 and NDCG@10 improves by approximately 13% point when using three-hop subgraphs compared to one-hop subgraphs. Regarding HowNet, the two-hop setting is optimal for both KGSF and C^2-CRS models as shown in Table 5.5.

Table 5.8 - 5.13 report the performance of four models on the conversation task. Similar to the case of the recommendation task, the optimal subgraph setting for the same knowledge graph on the same model varies across different datasets. Additionally, the amount of external knowledge required for a model to achieve the best performance on

Table 5.4: Recommendation task: experimental results on the TG-ReDial dataset with n-hop CN-DBpedia knowledge graphs.

	KBRD			**KGSF**			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
Recall@1	**0.011**	0.007	0.008	**0.011**	0.007	0.008	0.009	0.007	**0.010**
Recall@10	**0.059**	0.052	0.043	0.037	0.033	**0.039**	0.045	0.040	**0.052**
Recall@50	0.140	**0.149**	0.121	0.105	0.097	**0.112**	0.103	0.092	**0.112**
MRR@10	**0.024**	0.017	0.017	**0.018**	0.014	0.016	0.019	0.016	**0.020**
MRR@50	**0.027**	0.021	0.021	**0.021**	0.017	0.019	0.022	0.018	**0.023**
NDCG@10	**0.032**	0.025	0.023	**0.023**	0.018	0.021	0.025	0.022	**0.027**
NDCG@50	**0.049**	0.046	0.040	**0.037**	0.032	**0.037**	0.038	0.033	**0.041**

Table 5.5: Recommendation task: experimental results on the TG-ReDial dataset with n-hop HowNet knowledge graphs.

	KGSF			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
Recall@1	**0.005**	0.004	**0.005**	**0.006**	**0.006**	**0.006**
Recall@10	0.026	**0.032**	0.029	0.029	**0.032**	0.029
Recall@50	0.078	**0.079**	0.077	0.071	0.072	**0.076**
MRR@10	**0.011**	**0.011**	**0.011**	0.012	**0.014**	0.012
MRR@50	**0.013**	**0.013**	**0.013**	0.014	**0.015**	0.014
NDCG@10	0.014	**0.016**	0.015	0.016	**0.018**	0.016
NDCG@50	0.025	**0.026**	0.025	0.025	**0.027**	0.026

different conversation metrics may also differ. As presented in Table 5.8, using two-hop DBpedia subgraphs is reasonable for the KBRD, KGSF, and C^2-CRS models, while the three-hop setting is optimal for UniCRS if we are more interested in BLEU and ROUGE metrics. If inter-distinct metric is emphasized, we should introduce less external knowledge to KGSF and C^2-CRS models. Meanwhile, we are supposed to use larger DBpedia subgraphs for KBRD and UniCRS. The optimal DBpedia subgraph settings on the INSPIRED dataset are different from those on the Redial dataset. Table 5.12 indicates that the two-hop and three-hop settings are both potential options for the KBRD model on the INSPIRED dataset. KGSF performs best on BLEU and ROUGE metrics with one-hop subgraphs but produces more diverse outputs with three-hop subgraphs. For UniCRS, the amount of external knowledge required by different metrics varies. So, there is no specific recommended setting. A similar pattern is observed for the other three knowledge graphs: CN-DBpedia, ConceptNet, and HowNet. To be specific, the optimal setting of the amount of external knowledge is related to both the specific model and the specific metrics.

It is also worth mentioning that, based on our observations, there is no clear correlation between a model's optimal external knowledge setting for the recommendation task and that for the conversation task. Although we do not use a formal correlation

Table 5.6: Recommendation task: experimental results on the INSPIRED dataset with n-hop DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
Recall@1	0.036	**0.039**	0.026	**0.055**	0.049	0.049	0.043	**0.055**	0.043
Recall@10	0.104	0.100	**0.107**	**0.155**	0.149	0.123	0.137	**0.254**	0.148
Recall@50	**0.197**	0.191	0.168	0.285	**0.288**	0.256	0.316	**0.418**	0.340
MRR@10	0.056	**0.057**	0.053	**0.083**	0.072	0.071	0.069	**0.116**	0.075
MRR@50	**0.061**	**0.061**	0.056	**0.089**	0.078	0.077	0.078	**0.123**	0.083
NDCG@10	**0.068**	**0.068**	0.066	**0.100**	0.089	0.083	0.085	**0.149**	0.092
NDCG@50	**0.088**	0.086	0.079	**0.128**	0.119	0.113	0.125	**0.185**	0.133

Table 5.7: Recommendation task: experimental results on the INSPIRED dataset with n-hop ConceptNet knowledge graphs.

	KGSF		
	1 hop	2 hop	3 hop
Recall@1	0.068	0.058	**0.071**
Recall@10	0.142	0.133	**0.155**
Recall@50	0.236	**0.246**	0.233
MRR@10	0.087	0.082	**0.099**
MRR@50	0.091	0.088	**0.103**
NDCG@10	0.100	0.095	**0.113**
NDCG@50	0.120	0.119	**0.130**

metric, the experimental results suggest that the optimal configurations vary across the two tasks. We think that different factors influence model performance in each case. Take the impact of the DBpedia knowledge graph on the KBRD model as an example. On the ReDial dataset, the optimal subgraph setting for the recommendation task is three-hop, while the optimal setting for the conversation task is two-hop. It means that we should test on both recommendation and conversation tasks in order to determine the optimal amount of external knowledge.

5.3.2 Impact of incomplete knowledge graphs

In order to answer RQ2, we conduct experiments using four models with incomplete knowledge subgraphs. Table 5.14 - 5.19 summarize our experimental results on the recommendation task. We denote the settings where 50%, 30%, and 10% edges are randomly removed as '50%', '30%', and '10%' in the table header.

We evaluate the impact of incomplete knowledge graphs from two aspects: the sensitivity of model performance to changes in the percentage of removed edges and the horizontal comparison of model performance with n-hop experimental results. We first

Table 5.8: Conversation task: experimental results on the ReDial dataset with n-hop DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
BLEU@1	0.164	**0.177**	0.166	0.166	**0.174**	0.164	**0.240**	0.233	0.234	0.019	**0.020**	0.018
BLEU@2	0.038	**0.050**	0.043	0.038	**0.047**	0.036	**0.091**	0.085	0.084	**0.003**	**0.003**	**0.003**
BLEU@3	0.014	**0.024**	0.017	0.015	**0.026**	0.015	**0.048**	0.043	0.039	**0.002**	0.001	0.001
BLEU@4	0.008	**0.017**	0.009	0.009	**0.017**	0.009	**0.029**	0.026	0.021	**0.001**	0.000	0.000
ROUGE@1	0.256	**0.279**	0.260	0.260	**0.267**	0.253	**0.341**	0.331	0.327	0.053	**0.056**	0.053
ROUGE@2	0.045	**0.067**	0.063	0.049	**0.060**	0.044	**0.100**	0.093	0.090	**0.006**	**0.006**	0.005
ROUGE@L	0.250	**0.272**	0.256	0.254	**0.262**	0.249	**0.330**	0.321	0.317	0.052	**0.055**	0.052
Intra-dist@1	**0.874**	0.842	0.851	0.833	0.846	**0.849**	0.970	**0.971**	0.965	0.788	**0.799**	0.787
Intra-dist@2	0.996	0.973	**0.999**	0.976	0.983	**0.988**	**0.997**	0.996	**0.997**	**0.996**	0.992	**0.996**
Intra-dist@3	0.998	0.980	**0.999**	0.986	0.989	**0.992**	**0.935**	0.928	0.925	**0.998**	0.995	0.997
Intra-dist@4	**0.997**	0.971	0.930	0.976	**0.978**	0.966	**0.895**	0.883	0.887	0.772	**0.854**	0.787
Inter-dist@1	0.003	0.004	**0.006**	**0.009**	**0.009**	**0.009**	0.009	0.008	**0.011**	**0.035**	0.033	0.031
Inter-dist@2	0.012	0.013	**0.017**	**0.039**	0.034	0.033	0.036	0.032	**0.045**	**0.181**	0.171	0.156
Inter-dist@3	0.024	0.027	**0.030**	**0.080**	0.065	0.064	0.074	0.066	**0.091**	**0.326**	0.301	0.281
Inter-dist@4	0.042	**0.047**	0.044	**0.133**	0.106	0.101	0.125	0.111	**0.151**	**0.469**	0.418	0.410

analyze the performance of the KBRD model. We observe that KBRD's performance on most recommendation metrics is not sensitive to the percentage of removed edges. However, when we revisit KBRD's performance with n-hop knowledge graphs, we find it deteriorates with the incomplete knowledge graphs on the recommendation task. Especially on the INSPIRED dataset, the performance of KBRD with incomplete DBpedia subgraphs drops by more than 40% on most metrics. Our experimental results show that the sensitivity of KGSF's recommendation performance to incomplete knowledge graphs is related to datasets. On the ReDial and TG-ReDial datasets, KGSF's performance on most recommendation metrics remains relatively stable despite different ratios of removed edges. However, this observation does not hold for the INSPIRED dataset, where the performance of KGSF worsens with more missing DBpedia edges. Besides, the incompleteness of ConceptNet impacts KGSF's performance on different metrics to varying degrees. When comparing KGSF's performance using complete and incomplete knowledge subgraphs, we find that its performance on the TG-ReDial dataset and the ReDial dataset with incomplete ConceptNet subgraphs is close to its best performance with n-hop knowledge graphs. The UniCRS model is sensitive to the percentage of edges removed. Notably, even with 30% of edges removed from the DBpedia subgraph, its performance remains comparable to that with the 2-hop setting on the ReDial dataset. In contrast, the C^2-CRS model is generally not sensitive to the degree of knowledge graph incompleteness. Compared to the n-hop experimental results, its recommendation performance is worse with incomplete knowledge graphs in most cases. The only exception is using the incomplete HowNet knowledge graph on the TG-ReDial dataset. From Table 5.5 and Table 5.17, it is easy to see that the C^2-CRS's performance with 50% of

Table 5.9: Conversation task: experimental results on the ReDial dataset with n-hop ConceptNet knowledge graphs.

	KGSF			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
BLEU@1	0.164	0.165	**0.168**	0.021	**0.022**	**0.022**
BLEU@2	0.039	0.038	**0.041**	**0.003**	**0.003**	**0.003**
BLEU@3	0.018	0.015	**0.020**	**0.001**	**0.001**	**0.001**
BLEU@4	0.012	0.010	**0.013**	0.000	**0.001**	0.000
ROUGE@1	0.253	**0.257**	0.256	**0.055**	0.053	0.053
ROUGE@2	0.048	0.046	**0.049**	**0.006**	0.005	**0.006**
ROUGE@L	0.248	**0.252**	0.251	**0.054**	0.051	0.051
Intra-dist@1	0.833	0.846	**0.856**	**0.806**	0.794	0.804
Intra-dist@2	0.980	0.987	**0.992**	0.994	**0.996**	**0.996**
Intra-dist@3	0.986	0.992	**0.994**	0.996	0.997	**0.998**
Intra-dist@4	0.968	0.972	**0.982**	**0.863**	0.792	0.847
Inter-dist@1	**0.009**	**0.009**	**0.009**	**0.036**	0.033	0.031
Inter-dist@2	**0.037**	0.035	0.032	**0.201**	0.174	0.164
Inter-dist@3	**0.073**	0.068	0.060	**0.357**	0.312	0.297
Inter-dist@4	**0.119**	0.108	0.093	**0.491**	0.442	0.420

HowNet edges removed is not far from its best performance in n-hop knowledge graph experiments.

We report experimental results on the conversation task in Table 5.20 - 5.25. Our major finding is that knowledge graph incompleteness does not necessarily worsen models' performance on conversation metrics. This is similar to our findings for the recommendation task. Another main observation is that the ratio of removed edges affects model performance on inter-distinct differently than on other metrics in most cases. For the KBRD model, performance on the inter-distinct decreases as the degree of knowledge graph incompleteness increases on all datasets. Note that KBRD with the incomplete DBpedia knowledge graph achieves even higher inter-distinct scores than with the complete DBpedia knowledge graph. Conversely, KBRD's performance is not sensitive to the number of missing edges and is comparable to n-hop experimental results on other conversation metrics. The observation about KGSF's sensitivity on the recommendation task does not hold for the conversation task. In particular, changes in the degree of knowledge graph incompleteness on all datasets lead to fluctuations in KGSF's performance on the conversation task. Nevertheless, KGSF's performance on the ReDial dataset with incomplete ConceptNet knowledge graphs and on the INSPIRED dataset with incomplete DBpedia knowledge graphs is comparable to n-hop results. On the Redial dataset, the performance of the UniCRS model decreases to varying degrees across different conversation metrics as the proportion of removed knowledge graph edges changes. On the INSPIRED dataset, UniCRS performs best with a 30% edge removal ratio for all metrics except inter-distinct, which has an optimal setting of 50%. We also note that UniCRS achieves a higher inter-distinct score with the incomplete DBpedia knowledge

Table 5.10: Conversation task: experimental results on the TG-ReDial dataset with n-hop CN-DBpedia knowledge graphs.

	KBRD			**KGSF**			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
BLEU@1	0.283	0.288	**0.298**	0.271	**0.279**	0.271	0.158	**0.161**	0.150
BLEU@2	0.085	**0.102**	0.101	0.097	0.098	**0.099**	**0.035**	**0.035**	0.028
BLEU@3	0.028	0.034	**0.035**	0.031	0.032	**0.033**	**0.012**	0.011	0.009
BLEU@4	0.013	0.016	**0.018**	0.013	**0.014**	0.013	**0.005**	0.004	0.003
ROUGE@1	0.901	**0.907**	0.894	0.916	**0.922**	0.916	0.070	0.070	**0.075**
ROUGE@2	0.105	**0.116**	0.093	0.119	**0.131**	0.120	**0.000**	**0.000**	**0.000**
ROUGE@L	0.901	**0.907**	0.894	0.916	**0.922**	0.916	0.070	0.070	**0.075**
Intra-dist@1	0.690	0.703	**0.761**	0.648	**0.690**	0.669	0.805	**0.834**	0.817
Intra-dist@2	0.845	0.883	**0.930**	0.850	**0.887**	0.878	0.932	**0.964**	0.945
Intra-dist@3	0.908	0.942	**0.966**	0.922	**0.941**	0.939	0.948	**0.979**	0.963
Intra-dist@4	0.952	0.970	**0.981**	0.957	0.966	**0.967**	0.955	**0.986**	0.972
Inter-dist@1	0.010	**0.011**	0.010	0.015	**0.017**	0.016	0.032	0.033	**0.036**
Inter-dist@2	0.031	**0.033**	0.032	0.059	**0.067**	0.057	0.213	0.205	**0.241**
Inter-dist@3	0.065	**0.081**	0.071	0.134	**0.150**	0.131	0.381	0.359	**0.427**
Inter-dist@4	0.110	**0.148**	0.124	0.225	**0.247**	0.226	0.485	0.454	**0.536**

graph than with the complete one on the INSPIRED dataset. Finally, we examine the performance of the C^2-CRS model. The experimental results show that its performance is comparable to using the complete knowledge graphs, except for a decline observed in the experiments with the incomplete CN-DBpedia knowledge graph on the TG-ReDial dataset.

Table 5.11: Conversation task: experimental results on the TG-ReDial dataset with n-hop HowNet knowledge graphs.

	KGSF			**C^2-CRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
BLEU@1	0.285	0.279	**0.287**	0.152	**0.153**	0.151
BLEU@2	0.100	0.095	**0.105**	0.029	0.029	**0.031**
BLEU@3	0.033	0.030	**0.038**	0.008	0.008	**0.009**
BLEU@4	0.015	0.013	**0.017**	**0.003**	**0.003**	**0.003**
ROUGE@1	0.918	0.914	**0.919**	0.035	**0.044**	**0.044**
ROUGE@2	0.123	0.117	**0.126**	**0.000**	**0.000**	**0.000**
ROUGE@L	0.918	0.914	**0.919**	0.035	**0.044**	**0.044**
Intra-dist@1	0.701	0.690	**0.702**	0.814	0.825	**0.829**
Intra-dist@2	0.880	0.891	**0.898**	0.943	**0.961**	0.959
Intra-dist@3	0.930	0.943	**0.949**	0.963	**0.979**	0.974
Intra-dist@4	0.955	0.969	**0.972**	0.973	**0.986**	0.982
Inter-dist@1	**0.016**	**0.016**	0.015	0.029	0.029	**0.031**
Inter-dist@2	0.056	**0.058**	0.057	0.174	0.174	**0.184**
Inter-dist@3	0.124	**0.133**	0.126	0.311	0.307	**0.315**
Inter-dist@4	0.208	**0.227**	0.208	**0.395**	0.391	0.392

Table 5.12: Conversation task: experimental results on the INSPIRED dataset with n-hop DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**		
	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop	1 hop	2 hop	3 hop
BLEU@1	0.158	0.156	**0.161**	**0.159**	**0.159**	0.154	0.185	**0.189**	0.175
BLEU@2	0.036	0.034	**0.038**	**0.043**	0.041	0.036	0.054	**0.057**	0.048
BLEU@3	0.012	0.011	**0.017**	**0.018**	0.015	0.010	0.023	**0.025**	0.018
BLEU@4	0.006	0.006	**0.010**	**0.011**	0.008	0.006	**0.015**	0.014	0.009
ROUGE@1	0.237	**0.244**	0.237	**0.249**	**0.249**	0.234	**0.275**	**0.275**	0.270
ROUGE@2	0.036	0.036	**0.041**	**0.045**	0.042	0.036	0.051	**0.056**	0.050
ROUGE@L	0.225	**0.230**	0.223	**0.236**	0.234	0.221	0.256	**0.259**	0.253
Intra-dist@1	0.859	0.856	**0.874**	0.778	0.794	**0.822**	**0.920**	0.919	0.917
Intra-dist@2	0.978	0.980	**0.990**	0.928	0.945	**0.967**	**0.989**	0.986	0.988
Intra-dist@3	0.989	0.990	**0.995**	0.960	0.972	**0.986**	0.987	0.989	**0.994**
Intra-dist@4	0.993	0.995	**0.997**	0.975	0.985	**0.992**	0.961	**0.979**	0.971
Inter-dist@1	0.011	0.012	**0.014**	0.007	0.007	**0.008**	**0.067**	0.050	0.065
Inter-dist@2	0.039	**0.047**	0.043	0.029	0.029	**0.032**	**0.278**	0.189	0.271
Inter-dist@3	0.082	**0.103**	0.093	0.070	0.073	**0.080**	**0.502**	0.338	0.480
Inter-dist@4	0.138	**0.174**	0.158	0.135	0.137	**0.157**	**0.668**	0.470	0.646

Table 5.13: Conversation task: experimental results on the INSPIRED dataset with n-hop ConceptNet knowledge graphs.

	KGSF		
	1 hop	2 hop	3 hop
BLEU@1	0.160	**0.167**	0.155
BLEU@2	0.042	**0.044**	0.035
BLEU@3	**0.017**	0.013	0.008
BLEU@4	**0.009**	0.005	0.003
ROUGE@1	0.253	**0.281**	0.250
ROUGE@2	**0.046**	0.044	0.035
ROUGE@L	0.239	**0.258**	0.232
Intra-dist@1	**0.792**	0.739	0.767
Intra-dist@2	**0.943**	0.904	0.931
Intra-dist@3	**0.967**	0.953	0.964
Intra-dist@4	**0.981**	0.975	0.980
Inter-dist@1	**0.007**	0.005	0.006
Inter-dist@2	**0.030**	0.024	0.029
Inter-dist@3	**0.081**	0.065	0.076
Inter-dist@4	**0.155**	0.137	0.151

Table 5.14: Recommendation task: experimental results on the ReDial dataset with incomplete DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%	50%	30%	10%
Recall@1	**0.035**	0.034	0.033	**0.030**	**0.030**	0.028	0.032	**0.046**	0.014	0.049	**0.052**	0.051
Recall@10	0.171	**0.176**	0.174	0.173	**0.175**	0.174	0.161	**0.204**	0.074	0.209	**0.218**	0.212
Recall@50	0.323	0.332	**0.334**	**0.354**	**0.354**	0.353	0.364	**0.411**	0.136	0.379	0.380	**0.387**
MRR@10	0.068	**0.069**	0.068	**0.067**	**0.067**	0.066	0.065	**0.087**	0.030	0.090	**0.094**	0.093
MRR@50	0.075	**0.076**	**0.076**	**0.075**	**0.075**	0.074	0.074	**0.096**	0.033	0.098	**0.102**	0.101
NDCG@10	0.092	**0.094**	0.093	0.091	**0.092**	0.091	0.087	**0.114**	0.040	0.118	**0.123**	0.120
NDCG@50	0.125	0.128	**0.129**	**0.132**	0.131	0.131	0.132	**0.160**	0.054	0.155	**0.159**	**0.159**

Table 5.15: Recommendation task: experimental results on the ReDial dataset with incomplete ConceptNet knowledge graphs.

	KGSF			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%
Recall@1	**0.035**	0.034	**0.035**	0.050	**0.053**	0.052
Recall@10	**0.181**	0.179	0.177	0.227	0.221	**0.231**
Recall@50	0.359	**0.362**	**0.362**	0.406	0.400	**0.414**
MRR@10	**0.071**	0.070	**0.071**	0.096	0.096	**0.099**
MRR@50	**0.079**	0.078	**0.079**	0.105	0.105	**0.108**
NDCG@10	**0.096**	0.095	0.095	0.127	0.126	**0.130**
NDCG@50	0.136	0.135	**0.137**	0.167	0.166	**0.170**

Table 5.16: Recommendation task: experimental results on the TG-ReDial dataset with incomplete CN-DBpedia knowledge graphs.

	KBRD			**KGSF**			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%
Recall@1	**0.008**	0.007	0.006	**0.009**	0.006	0.008	**0.005**	0.004	0.004
Recall@10	0.044	0.047	**0.050**	0.044	0.045	**0.048**	**0.049**	0.043	0.045
Recall@50	0.102	0.126	**0.137**	0.116	**0.117**	0.106	0.097	0.097	**0.101**
MRR@10	**0.018**	0.017	0.017	**0.018**	0.016	**0.018**	**0.017**	0.014	0.014
MRR@50	**0.021**	**0.021**	0.020	**0.021**	0.019	0.020	**0.019**	0.016	0.017
NDCG@10	0.024	0.024	**0.025**	0.024	0.023	**0.025**	**0.025**	0.020	0.021
NDCG@50	0.037	0.041	**0.043**	**0.039**	0.038	0.037	**0.035**	0.032	0.034

Table 5.17: Recommendation task: experimental results on the TG-ReDial dataset with incomplete HowNet knowledge graphs.

	KGSF			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%
Recall@1	**0.007**	0.003	0.005	**0.007**	0.005	0.005
Recall@10	0.027	0.025	**0.030**	**0.033**	0.029	0.027
Recall@50	0.074	0.066	**0.076**	**0.081**	0.076	0.068
MRR@10	**0.012**	0.008	**0.012**	**0.013**	0.011	0.011
MRR@50	**0.014**	0.010	**0.014**	**0.015**	0.013	0.013
NDCG@10	**0.016**	0.012	**0.016**	**0.017**	0.015	0.015
NDCG@50	0.025	0.021	**0.026**	**0.028**	0.026	0.023

Table 5.18: Recommendation task: experimental results on the INSPIRED dataset with incomplete DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%
Recall@1	**0.029**	**0.029**	**0.029**	0.039	**0.055**	**0.055**	**0.043**	**0.043**	**0.043**
Recall@10	0.039	**0.052**	0.045	0.133	0.129	**0.139**	0.137	0.148	**0.152**
Recall@50	0.081	**0.087**	0.084	0.291	0.275	**0.298**	0.340	**0.352**	0.348
MRR@10	0.031	**0.036**	0.032	0.069	0.075	**0.080**	0.069	**0.075**	**0.075**
MRR@50	0.033	**0.037**	0.033	0.076	0.082	**0.087**	0.079	0.083	**0.084**
NDCG@10	0.033	**0.040**	0.035	0.085	0.088	**0.094**	0.085	0.092	**0.093**
NDCG@50	0.042	**0.047**	0.043	0.118	0.119	**0.128**	0.129	**0.136**	**0.136**

Table 5.19: Recommendation task: experimental results on the INSPIRED dataset with incomplete ConceptNet knowledge graphs.

	KGSF		
	50%	30%	10%
Recall@1	0.058	**0.061**	0.042
Recall@10	**0.152**	0.146	0.133
Recall@50	0.220	**0.233**	0.223
MRR@10	0.085	**0.087**	0.069
MRR@50	0.088	**0.090**	0.073
NDCG@10	**0.101**	**0.101**	0.084
NDCG@50	0.116	**0.119**	0.104

Table 5.20: Conversation task: experimental results on the ReDial dataset with incomplete DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%	50%	30%	10%
BLEU@1	**0.175**	0.165	0.163	**0.170**	0.163	0.168	**0.236**	0.230	0.232	**0.021**	0.018	0.017
BLEU@2	**0.049**	0.044	0.042	**0.044**	0.039	0.043	0.082	0.082	**0.086**	**0.003**	0.002	0.002
BLEU@3	**0.023**	0.019	0.018	**0.023**	0.017	**0.023**	0.041	0.041	**0.043**	**0.001**	**0.001**	**0.001**
BLEU@4	**0.014**	0.011	0.011	**0.015**	0.011	0.014	0.024	0.024	**0.025**	0.000	0.000	**0.001**
ROUGE@1	**0.287**	0.274	0.273	**0.258**	**0.258**	0.256	**0.333**	0.325	0.329	0.051	**0.052**	0.046
ROUGE@2	**0.065**	0.056	0.057	0.054	0.048	**0.056**	0.091	0.089	**0.093**	**0.006**	0.004	0.005
ROUGE@L	**0.281**	0.268	0.266	**0.253**	0.252	0.251	**0.322**	0.315	0.318	0.050	**0.051**	0.045
Intra-dist@1	**0.864**	0.796	0.821	**0.846**	0.843	0.844	0.966	**0.976**	0.971	**0.801**	0.794	0.774
Intra-dist@2	**0.982**	0.955	0.957	0.984	0.982	**0.986**	0.995	**0.997**	0.996	0.997	0.995	**0.998**
Intra-dist@3	**0.986**	0.971	0.966	0.987	0.988	**0.989**	**0.950**	0.929	0.914	0.998	0.997	**0.999**
Intra-dist@4	**0.989**	0.908	0.947	**0.975**	0.949	0.966	**0.914**	0.890	0.863	0.817	**0.821**	0.725
Inter-dist@1	0.003	**0.004**	0.003	**0.011**	0.008	0.010	0.006	**0.009**	0.008	0.033	**0.037**	0.032
Inter-dist@2	0.010	**0.013**	0.012	**0.042**	0.027	0.038	0.027	**0.037**	0.032	0.156	**0.185**	0.142
Inter-dist@3	0.017	**0.024**	**0.024**	**0.075**	0.052	0.067	0.063	**0.075**	0.065	0.275	**0.325**	0.251
Inter-dist@4	0.026	0.038	**0.042**	**0.112**	0.082	0.099	0.118	**0.125**	0.107	0.384	**0.455**	0.371

Table 5.21: Conversation task: experimental results on the ReDial dataset with incomplete ConceptNet knowledge graphs.

	KGSF			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%
BLEU@1	0.171	**0.172**	0.167	0.018	**0.022**	0.016
BLEU@2	0.043	**0.046**	0.040	0.003	**0.004**	0.002
BLEU@3	0.022	**0.024**	0.018	**0.001**	**0.001**	**0.001**
BLEU@4	0.015	**0.016**	0.013	**0.001**	**0.001**	0.000
ROUGE@1	**0.267**	0.265	0.256	0.046	**0.060**	0.044
ROUGE@2	0.054	**0.060**	0.049	0.005	**0.008**	0.004
ROUGE@L	**0.261**	0.260	0.251	0.045	**0.058**	0.043
Intra-dist@1	**0.852**	0.841	0.848	0.789	**0.803**	0.784
Intra-dist@2	0.983	0.985	**0.987**	0.996	0.995	**0.998**
Intra-dist@3	0.988	0.989	**0.991**	0.997	0.997	**0.999**
Intra-dist@4	**0.979**	0.949	0.974	0.789	**0.820**	0.772
Inter-dist@1	0.007	**0.012**	0.010	**0.034**	0.030	**0.034**
Inter-dist@2	0.028	**0.042**	0.037	**0.190**	0.167	0.163
Inter-dist@3	0.056	**0.074**	0.069	**0.348**	0.308	0.286
Inter-dist@4	0.092	**0.108**	0.106	**0.497**	0.439	0.409

Table 5.22: Conversation task: experimental results on the TG-ReDial dataset with incomplete CN-DBpedia knowledge graphs.

	KBRD			**KGSF**			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%
BLEU@1	0.262	**0.298**	0.277	0.275	0.275	**0.277**	**0.159**	0.150	0.147
BLEU@2	0.086	**0.108**	0.092	0.087	0.093	**0.097**	**0.032**	0.031	0.027
BLEU@3	0.031	**0.038**	0.029	0.022	0.028	**0.032**	**0.010**	**0.010**	0.008
BLEU@4	0.014	**0.017**	0.014	0.008	0.012	**0.013**	**0.004**	**0.004**	0.003
ROUGE@1	0.902	**0.906**	0.901	0.915	**0.919**	0.918	**0.084**	0.066	0.027
ROUGE@2	0.108	**0.115**	0.105	0.118	**0.126**	0.123	**0.000**	**0.000**	**0.000**
ROUGE@L	0.902	**0.906**	0.901	0.915	**0.919**	0.918	**0.084**	0.066	0.027
Intra-dist@1	0.642	**0.728**	0.686	0.678	0.666	**0.684**	0.802	0.811	**0.819**
Intra-dist@2	0.800	**0.903**	0.872	0.871	0.855	**0.881**	0.936	0.941	**0.946**
Intra-dist@3	0.858	**0.949**	0.927	0.934	0.920	**0.940**	0.954	0.960	**0.963**
Intra-dist@4	0.894	**0.972**	0.954	0.963	0.954	**0.968**	0.964	0.969	**0.972**
Inter-dist@1	0.007	0.008	**0.009**	**0.017**	0.015	0.015	0.034	**0.035**	0.034
Inter-dist@2	0.021	0.026	**0.028**	**0.061**	0.057	0.055	0.202	**0.228**	0.195
Inter-dist@3	0.049	0.060	**0.068**	**0.138**	0.130	0.124	0.356	**0.402**	0.330
Inter-dist@4	0.086	0.108	**0.126**	**0.231**	0.221	0.211	0.450	**0.508**	0.412

Table 5.23: Conversation task: experimental results on the TG-ReDial dataset with incomplete HowNet knowledge graphs.

	KGSF			**C^2-CRS**		
	50%	30%	10%	50%	30%	10%
BLEU@1	**0.291**	0.283	0.287	0.160	0.155	**0.164**
BLEU@2	**0.108**	0.100	0.096	0.033	0.033	**0.037**
BLEU@3	**0.039**	0.032	0.027	0.009	0.009	**0.012**
BLEU@4	**0.017**	0.015	0.010	0.004	0.003	**0.005**
ROUGE@1	**0.923**	0.920	0.922	0.036	**0.059**	0.058
ROUGE@2	**0.132**	0.127	0.131	**0.000**	**0.000**	**0.000**
ROUGE@L	**0.923**	0.920	0.922	0.036	**0.059**	0.058
Intra-dist@1	0.715	0.693	**0.728**	0.824	**0.833**	0.822
Intra-dist@2	0.907	0.894	**0.908**	**0.964**	0.960	0.955
Intra-dist@3	0.952	0.948	**0.956**	**0.979**	0.977	0.973
Intra-dist@4	0.972	0.972	**0.977**	**0.986**	0.985	0.982
Inter-dist@1	0.014	0.011	**0.016**	0.028	0.030	**0.032**
Inter-dist@2	0.052	0.042	**0.057**	0.166	0.173	**0.200**
Inter-dist@3	0.115	0.100	**0.129**	0.293	0.293	**0.358**
Inter-dist@4	0.194	0.178	**0.217**	0.372	0.366	**0.461**

Table 5.24: Conversation task: experimental results on the INSPIRED dataset with incomplete DBpedia knowledge graphs.

	KBRD			**KGSF**			**UniCRS**		
	50%	30%	10%	50%	30%	10%	50%	30%	10%
BLEU@1	**0.163**	0.162	0.162	0.161	0.149	**0.163**	0.170	**0.187**	0.177
BLEU@2	**0.039**	**0.039**	**0.039**	**0.040**	0.030	0.037	0.045	**0.057**	0.041
BLEU@3	0.014	0.014	**0.017**	**0.015**	0.011	0.013	0.014	**0.025**	0.016
BLEU@4	0.007	**0.010**	**0.010**	**0.009**	0.006	0.007	0.006	**0.015**	0.009
ROUGE@1	**0.243**	0.242	0.238	0.241	0.225	**0.243**	0.264	**0.276**	0.264
ROUGE@2	**0.041**	0.038	0.040	**0.040**	0.033	0.037	0.044	**0.057**	0.040
ROUGE@L	**0.227**	**0.227**	0.226	0.227	0.213	**0.231**	0.247	**0.259**	0.248
Intra-dist@1	0.829	**0.865**	0.854	0.823	**0.848**	0.792	0.926	0.926	**0.938**
Intra-dist@2	0.967	**0.985**	0.969	0.968	**0.982**	0.934	0.990	0.987	**0.992**
Intra-dist@3	0.985	**0.994**	0.977	0.987	**0.991**	0.963	0.982	0.984	**0.990**
Intra-dist@4	0.991	**0.997**	0.983	0.994	**0.995**	0.977	0.953	0.963	**0.982**
Inter-dist@1	0.012	0.013	**0.014**	0.008	**0.009**	0.008	**0.071**	0.049	0.047
Inter-dist@2	0.042	0.046	**0.052**	0.031	**0.038**	0.035	**0.304**	0.181	0.185
Inter-dist@3	0.095	0.101	**0.109**	0.077	**0.096**	0.085	**0.537**	0.310	0.333
Inter-dist@4	0.166	0.178	**0.182**	0.151	**0.182**	0.153	**0.703**	0.429	0.474

Table 5.25: Conversation task: experimental results on the INSPIRED dataset with incomplete ConceptNet knowledge graphs.

	KGSF		
	50%	30%	10%
BLEU@1	0.150	**0.151**	**0.151**
BLEU@2	**0.036**	0.034	0.032
BLEU@3	**0.011**	0.010	0.010
BLEU@4	**0.006**	0.005	**0.006**
ROUGE@1	0.240	**0.242**	0.235
ROUGE@2	**0.036**	0.035	0.032
ROUGE@L	0.227	**0.228**	0.221
Intra-dist@1	0.808	0.791	**0.841**
Intra-dist@2	0.958	0.941	**0.971**
Intra-dist@3	0.980	0.969	**0.984**
Intra-dist@4	0.988	0.984	**0.991**
Inter-dist@1	0.008	0.007	**0.009**
Inter-dist@2	0.033	0.032	**0.037**
Inter-dist@3	0.085	0.086	**0.090**
Inter-dist@4	0.167	**0.175**	0.161

Chapter 6

Discussion

The knowledge graph is an important source of external knowledge in the design of CRSs. However, existing studies have not quantitatively examined how the amount of knowledge introduced by the knowledge graph affects the performance of CRSs. Our project investigates the role of knowledge graphs in CRSs from two perspectives: varying amounts of external knowledge and different degrees of knowledge graph incompleteness. The performance of CRSs is evaluated in both recommendation and conversation tasks. This chapter discusses the results of our experiments and answers two research questions presented earlier.

6.1 RQ1: How do amounts of external knowledge affect the performance of CRSs?

Considering the n-hop experimental results comprehensively, we find that using larger knowledge subgraphs does not necessarily improve model performance. In fact, an excessive amount of external knowledge may impair model performance in both the recommendation and conversation tasks. We think the reason for this phenomenon is the size of conversational recommendation datasets is generally small and incorporating a large knowledge graph brings the risk of introducing too much noise. It ultimately results in the benefits of integrating external knowledge being offset by the introduced noise. Another major finding is that the amount of external knowledge required for the model to achieve optimal performance on different metrics may be different. Besides, there is no obvious correlation between the optimal settings for the recommendation task and those for the conversation task. For any specific dataset, the optimal amount

of external knowledge needs to be determined through extensive experiments. We note that existing research does not describe the process of creating knowledge subgraphs in detail and their code implementations directly import the processed knowledge graph data files. Moreover, they use the same knowledge graph in the recommendation and conversation tasks due to limitations in the existing CRS architecture. Our experimental results suggest that a potential direction for improving CRS performance is to design separate external knowledge integration modules for recommendation and conversation tasks.

Our experiments adopt a comprehensive set of evaluation metrics to examine the performance of CRSs. Unlike some studies that use only recall and inter-distinct scores, our experimental results cover commonly used recommendation and conversation metrics. We find that the amount of external knowledge required for a model to achieve the best performance on different metrics usually varies. Therefore, we need to balance the importance of different metrics based on the business scenario to determine the optimal external knowledge setting in practical applications.

6.2 RQ2: How does the incompleteness of knowledge graphs affect the performance of CRSs?

Our experimental results demonstrate that the incompleteness of knowledge graphs does not have to lead to a decline in model performance. Actually, the incompleteness of the knowledge graph sometimes slightly improves the model's performance compared to n-hop experimental results. We do not observe any general pattern regarding the impact of the degree of knowledge graph incompleteness on model performance. Similar to the external knowledge amount settings in the n-hop experiments, it is necessary to conduct experiments to determine the percentage of missing knowledge graph edges that minimizes the impact on model performance for a specific model and dataset. In addition, we notice that all four models are not sensitive to changes in the degree of knowledge graph incompleteness in some experiments. These results support our suggestion in Section 6.1 that designing separate external knowledge fusion modules for recommendation and conversation tasks may enhance the performance of CRSs. In practice, we can also consider reducing the size of the knowledge graph used in CRSs if the model's performance does not degrade when the knowledge graph is incomplete. This approach is beneficial because a smaller knowledge graph can effectively reduce the model training time and the consumption of computational resources.

6.3 Limitations and future work

In this section, we summarize the limitations of our study and suggest directions for future work.

Data. Our experiments cover three public conversational recommendation datasets. In future work, we can examine the role of knowledge graphs in model performance on more datasets. However, we anticipate that generalizing the model to new datasets will require significant code refactoring efforts, since the code implementations provided by paper authors often suffer from poor encapsulation and lack of project documentation. Compared to datasets from other directions in the recommender systems field, the scale of conversational recommendation datasets is relatively small. In addition, existing models have not been deployed online in real-world application scenarios, resulting in a lack of performance statistics on online data. We think a potential solution is to explore the possibility of applying user simulators in augmenting conversational recommendation data in future studies.

Generalizability. Due to computing power limitations, we mainly focus on four representative CRSs in our project. A future research direction is to conduct experiments related to the role of knowledge graphs on more CRSs. Additionally, we mainly introduce the information of knowledge graphs through n-hop subgraphs in experiments, which is consistent with the existing work. However, we think trying other alternative graph traversal methods, such as the PageRank algorithm, to integrate knowledge graphs is also a direction worth studying.

Evaluation. We evaluate the model performance on commonly used recommendation and conversation metrics in our project. This bridges the research gap in some papers that only report recall and inter-distinct scores. However, we do not use evaluation metrics that require human participation due to resource constraints. Meanwhile, we note that there have been advancements in learning-based methods in the field of dialogue system evaluation. In future research, we can consider introducing this kind of evaluation system into the evaluation of CRSs to improve the accuracy and comprehensiveness of evaluations.

Chapter 7

Conclusion

In this research project, we investigated the performance of CRSs from a reproducibility perspective. Specifically, we examined the impact of knowledge graphs on CRS performance, which had not been quantitatively studied in existing work. Our two research questions focused on the amount of information brought by knowledge graphs and the possibility of missing edges in knowledge graphs, respectively. To address RQ1, we controlled the amount of external knowledge by adjusting the size of the knowledge subgraphs. We found that the optimal knowledge graph settings varied across different datasets and metrics for a specific CRS model. Notably, an excess of external knowledge could impair model performance. To answer RQ2, we simulated the incomplete knowledge graphs by randomly removing edges. Our main finding is that incomplete knowledge graphs do not necessarily lead to a decline in model performance. Experimental results suggest that a possible improvement to the current CRS architecture is to integrate knowledge graphs through two separate modules for recommendation and conversation tasks. For models whose performance does not deteriorate with knowledge graph incompleteness, we can consider reducing the size of knowledge graphs to accelerate the model training process in real-world applications. Due to resource limitations, our experiments cover only four models and three datasets. Future studies can generalize our methodology to more CRSs and datasets. It is also interesting to try different graph traversal strategies to construct knowledge subgraphs and evaluate the system with more sophisticated approaches.

Bibliography

[1] Oren Barkan and Noam Koenigstein. Item2vec: Neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2016.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[3] Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. Training and testing low-degree polynomial data mappings via linear svm. *J. Mach. Learn. Res.*, 11:1471–1490, aug 2010.

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

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, pages 7–10, New York, NY, USA, 2016. Association for Computing Machinery.

[6] Zhendong Dong and Qiang Dong. Hownet - a hybrid language and knowledge resource. In *International Conference on Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003*, pages 820–824, 2003.

[7] Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):81–173, 2011.

[8] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126, 2021.

[9] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, dec 1992.

[10] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 855–864, New York, NY, USA, 2016. Association for Computing Machinery.

[11] Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. INSPIRED: Toward sociable recommendation dialog systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8142–8152, Online, November 2020. Association for Computational Linguistics.

[12] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 355–364, New York, NY, USA, 2017. Association for Computing Machinery.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 173–182, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee.

[14] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. A survey on conversational recommender systems. *ACM Comput. Surv.*, 54(5), May 2021.

[15] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514, 2022.

[16] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 43–50, New York, NY, USA, 2016. Association for Computing Machinery.

[17] Mojtaba Komeili, Kurt Shuster, and Jason Weston. Internet-augmented dialogue generation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[18] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[19] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. Dbpedia – a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195, 2015. 2.

[20] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California, June 2016. Association for Computational Linguistics.

[21] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany, August 2016. Association for Computational Linguistics.

[22] Raymond Li, Samira Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 9748–9758, Red Hook, NY, USA, 2018. Curran Associates Inc.

[23] Shuokai Li, Ruobing Xie, Yongchun Zhu, Xiang Ao, Fuzhen Zhuang, and Qing He. User-centric conversational recommendation with multi-aspect user modeling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 223–233, New York, NY, USA, 2022. Association for Computing Machinery.

[24] Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 733–743, 2017.

[25] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[26] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[27] Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. RevCore: Review-augmented conversational recommendation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1161–1173, Online, August 2021. Association for Computational Linguistics.

[28] Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. CR-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1839–1851, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[29] Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. ParlAI: A dialog research software platform. In Lucia Specia, Matt Post, and Michael Paul, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[30] Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, and Erik Cambria. Recent advances in deep learning based dialogue systems: a systematic survey. *Artificial Intelligence Review*, 56(4):3055–3155, Apr 2023.

[31] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, USA, 2002. Association for Computational Linguistics.

[32] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, 2014. Association for Computing Machinery.

[33] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1149–1154, 2016.

[34] Zhaochun Ren, Zhi Tian, Dongdong Li, Pengjie Ren, Liu Yang, Xin Xin, Huasheng Liang, Maarten de Rijke, and Zhumin Chen. Variational reasoning about user preferences for conversational recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 165–175, New York, NY, USA, 2022. Association for Computing Machinery.

[35] Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000, 2010.

[36] Deepjyoti Roy and Mala Dutta. A systematic review and research perspective on recommender systems. *Journal of Big Data*, 9(1):59, May 2022.

[37] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 111–112, New York, NY, USA, 2015. Association for Computing Machinery.

[38] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and JC Mao. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 255–262, New York, NY, USA, 2016. Association for Computing Machinery.

[39] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In Chengqing Zong and Michael Strube, editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586, Beijing, China, July 2015. Association for Computational Linguistics.

[40] Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, Morteza Behrooz, William Ngan, Spencer Poff, Naman Goyal, Arthur Szlam, Y-Lan Boureau, Melanie Kambadur, and Jason Weston. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage, 2022.

[41] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: an open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 4444–4451. AAAI Press, 2017.

[42] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[43] Hongru Wang, Lingzhi Wang, Yiming Du, Liang Chen, Jingyan Zhou, Yufei Wang, and Kam-Fai Wong. A survey of the evolution of language model-based dialogue systems, 2023.

[44] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 417–426, New York, NY, USA, 2018. Association for Computing Machinery.

[45] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 839–848, New York, NY, USA, 2018. Association for Computing Machinery.

[46] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pages 1929–1937, New York, NY, USA, 2022. Association for Computing Machinery.

[47] Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. In Mirella Lapata, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain, April 2017. Association for Computational Linguistics.

[48] Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. Cn-dbpedia: A never-ending chinese knowledge extraction system. In Salem Benferhat, Karim Tabia, and Moonis Ali, editors, *Advances in Artificial Intelligence: From Theory to Practice*, pages 428–438, Cham, 2017. Springer International Publishing.

[49] Bowen Yang, Cong Han, Yu Li, Lei Zuo, and Zhou Yu. Improving conversational recommendation systems' quality with context-aware item meta-information. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 38–48, Seattle, United States, July 2022. Association for Computational Linguistics.

[50] Jiayi Zhang, Chongyang Tao, Zhenjing Xu, Qiaojing Xie, Wei Chen, and Rui Yan. Ensemblegan: Adversarial learning for retrieval-generation ensemble model on short-text conversation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 435–444, New York, NY, USA, 2019. Association for Computing Machinery.

[51] Weinan Zhang, Tianming Du, and Jun Wang. Deep learning over multi-field categorical data. In *Advances in Information Retrieval*, pages 45–57, Cham, 2016. Springer International Publishing.

[52] Xiaoyu Zhang, Xin Xin, Dongdong Li, Wenxuan Liu, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. Variational reasoning over incomplete knowledge graphs for conversational recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 231–239, 2023.

[53] Yongfeng Zhang and Xu Chen. Explainable recommendation: A survey and new perspectives. *Found. Trends Inf. Retr.*, 14(1):1–101, March 2020.

[54] Tiancheng Zhao and Maxine Eskenazi. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In Raquel Fernandez, Wolfgang Minker, Giuseppe Carenini, Ryuichiro Higashinaka, Ron Artstein, and Alesia Gainer, editors, *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–10, Los Angeles, September 2016. Association for Computational Linguistics.

[55] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 167–176, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

[56] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. Deep interest evolution network for click-through rate prediction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019.

[57] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 1059–1068, New York, NY, USA, 2018. Association for Computing Machinery.

[58] Jinfeng Zhou, Bo Wang, Ruifang He, and Yuexian Hou. CRFR: Improving conversational recommender systems via flexible fragments reasoning on knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4324–4334, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[59] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 1006–1014, New York, NY, USA, 2020. Association for Computing Machinery.

[60] Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4128–4139, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[61] Yuanhang Zhou, Kun Zhou, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and He Hu. C^2-crs: Coarse-to-fine contrastive learning for conversational recommender system. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, pages 1488–1496, New York, NY, USA, 2022. Association for Computing Machinery.