Algerian Democratic and Popular Republic
Ministry of Higher Education and Scientific Research
KASDI MERBAH UNIVERSITY OUARGLA
Faculty of New Information and Communication Technologies
Department of Computer Science and Information Technology

Thesis Submitted in Candidacy for a Master Degree in computer science, option

Artificial Intelligence and Data Science

by: Samah Maamri & Wissal Benmessaoud

Theme

Item-based Knowledge Graph for Understanding User Intent in Session-Based Recommendation Systems

Evaluation date: /06/2023

Jury Members:

Dr.		Jury Chair	UKM Ouargla
Dr.	Khadra Bouanane	Supervisor	UKM Ouargla
Dr.	Meriem Korichi	Co-Supervisor	UKM Ouargla
Dr.		Examiner	UKM Ouargla

Academic year: 2023/2024

Acknowledgments

We begin by expressing our deepest gratitude to Allah, the Most Merciful and Most Compassionate, for His guidance and blessings throughout the completion of our Master's degree thesis.

*We extend heartfelt thanks to our supervisors, **Dr. Bouanane Khadra** and **Dr. Korichi Meriem**, for their invaluable guidance and unwavering support during the research process.*

*We also appreciate the faculty and staff at **The Department of Computer Science and Information Technology** for fostering an environment of academic excellence.*

To our beloved parents, we express profound gratitude for their unconditional love, unwavering support, and sacrifices. Your belief in our potential has been a constant source of strength.

Finally, we are sincerely grateful to all who contributed to the successful completion of our thesis. Your assistance, encouragement, and leadership have been invaluable.

Thank you all

Samah Maamri & Wissal Benmessaoud

Dedication

To embark on this profound journey that has consumed a significant portion of my academic life, I am deeply indebted to the divine guidance of Allah. Therefore, I begin by expressing my utmost gratitude to Allah.

I extend my heartfelt appreciation to my beloved mother and my ever-supportive father, whose unwavering love and encouragement have been my guiding light throughout my educational years. They have stood by me through every triumph and tribulation, sharing in the joys and sorrows of my educational pursuits.

I am also grateful to my siblings, whose support and companionship have been a constant source of strength and inspiration.

*I would like to express my profound gratitude to my partner in this thesis, **Samah**, whose dedication and support have been invaluable in completing this work.*

*To my dear friends **Bouchra**, **Amira**, and **Oumaima**, your friendship and encouragement have meant a lot to me.*

With deep gratitude and profound appreciation, I dedicate this work to all those who have played a part in shaping my path. May their unwavering support and belief in me be forever etched in my heart as a reminder of the transformative power of love and encouragement.

Wissal Benmessaoud

Dedication

To my beloved family and friends,

I want to express my deepest gratitude and love to each of you who has supported and accompanied me throughout my life. To my parents,You are my pillars of strength and wisdom. Thank you for your unconditional love, sacrifices, and invaluable guidance. ***Mom****, your tenderness and patience are my source of comfort.* ***Dad****, your dedication and wisdom inspire me every day. You've shaped me into the person I am today, and I'm forever grateful. To my brothers,* ***Oussama*** *and* ***Chawki,****You're not just my brothers but my lifelong friends. Your humor and determination have enriched my life in countless ways. To my sisters,* ***Chaima****,* ***Marwa,Safa*** *and* ***Amel,****You're my confidantes and cheerleaders. Your presence, laughter, and advice bring me endless joy and strength. To* ***my aunts, uncles,*** *and* ***grandmother****,Your love and support have been a constant blessing. Your wisdom and stories have shaped my perspectives and enriched my life. To all my family,Thank you for the love, laughter, and unwavering support that have shaped me into who I am today. To my dear friends,* ***Oumaima, Wissal, Bouchra****, and* ***Amira****,You're my chosen family. Your friendship, support, and shared moments mean the world to me.*

With all my love,
Samah Maamri

Abstract

Recommendation systems are crucial for improving user experiences through personalized suggestions based on predicted preferences. Session-based recommendation systems focus on predicting the next item a user is likely to interact with based on their current session activities. This research introduces a novel session-based next-item recommendation model that utilizes Knowledge Graphs (KGs) to uncover user intents within sessions. Existing recommendation models often struggle with basic relational modeling, limiting their ability to capture detailed user-item relationships and long-term connections. Our proposed approach, inspired by hierarchical semantic networks, addresses these limitations by integrating KGs to better understand user intents and improve intra-session dependency modeling. By leveraging item auxiliary KGs, our model provides more precise and personalized recommendations. We assess the effectiveness of our approach in terms of recommendation accuracy and personalization, demonstrating the model's capability to provide explainability for its recommendations.

Keywords: Session-based recommendation systems, Knowledge Graphs, Graph Convolutional Networks, Explainability, Personalization.

ملخص

الأنظمة القائمة على التوصيات هي حاسمة لتحسين تجارب المستخدمين من خلال الاقتراحات الشخصية المستندة إلى التفضيلات المتوقعة. تركز أنظمة التوصيات قائمة الجلسة Session على تنبؤ العنصر التالي الذي قد يتفاعل معه المستخدم بناءً على نشاطات جلسته الحالية. يقدم هذا البحث نموذجًا جديدًا لتوصية العناصر التالية في الجلسة يستخدم الرسوم البيانية للمعرفة (Knowledge Graphs) للكشف عن نوايا المستخدمين داخل الجلسات. النماذج التقليدية للتوصيات غالبًا ما تعاني من النمذجة العلاقاتية الأساسية، مما يحد من قدرتها على التقاط العلاقات التفصيلية بين المستخدم والعنصر والاتصالات طويلة الأمد. يعالج النهج المقترح، المستوحى من hierarchical semantic networks ، هذه القيود من خلال دمج Knowledge Graphs لفهم نوايا المستخدمين بشكل أفضل وتحسين نمذجة الاعتمادية داخل الجلسة. من خلال استغلال Knowledge Graphs المساعدة للعناصر، يوفر نموذجنا توصيات أكثر دقة وشخصية. نقيم فعالية نهجنا من خلال دقة التوصية والتخصيص الشخصي، مع توضيح قدرة النموذج على توفير قابلية التفسير لتوصياته.

الكلمات المفتاحية: أنظمة التوصية القائمة على الجلسات، Knowledge Graphs ، Graph Convolutional Networks ، التفسير، التخصيص، Session-based recommendation systems .

Résumé

Les systèmes de recommandation sont cruciaux pour améliorer l'expérience utilisateur en proposant des suggestions personnalisées basées sur les préférences prédites. Les systèmes de recommandation basés sur la session se concentrent sur la prédiction du prochain élément avec lequel un utilisateur est susceptible d'interagir en fonction de ses activités de session actuelles. Cette recherche présente un nouveau modèle de recommandation d'élément suivant basé sur la session qui utilise des Graphes de Connaissances (KGs) pour découvrir les intentions des utilisateurs au sein des sessions. Les modèles de recommandation existants ont souvent du mal avec la modélisation relationnelle de base, ce qui limite leur capacité à capturer les relations détaillées entre l'utilisateur et l'élément ainsi que les connexions à long terme. Notre approche proposée, inspirée des réseaux sémantiques hiérarchiques, adresse ces limitations en intégrant les KGs pour mieux comprendre les intentions des utilisateurs et améliorer la modélisation des dépendances intra-session. En exploitant les KGs auxiliaires des éléments, notre modèle fournit des recommandations plus précises et personnalisées. Nous évaluons l'efficacité de notre approche en termes de précision des recommandations et de personnalisation, démontrant la capacité du modèle à fournir une explication pour ses recommandations.

Mots-clés: Systèmes de recommandation, Recommandations basées sur les sessions, Graphes de Connaissances, Réseaux de Convolution sur Graphe, Explicabilité.

Contents

List of Figures

List of Tables

General Introduction

The immense volume of data generated by internet users presents a significant challenge in finding relevant information efficiently. As a result, recommendation systems have become vital in guiding users through this digital expanse, aiming to predict preferences and offer personalized suggestions, thereby enhancing user experience and engagement [48, 9]. This thesis explores the integration of Knowledge Graphs (KGs) and Graph Convolutional Networks (GCNs) within Session-Based Recommendation Systems (SBRS) to tackle the challenges posed by dynamic user preferences and sparse data environments [60, 26].

Recommendation systems have evolved from early collaborative filtering and content-based approaches to sophisticated deep learning models capable of capturing intricate user behaviors. Traditional methods depended on historical user-item interactions to infer preferences. However, advancements in machine learning, particularly GCNs, have revolutionized these systems by using graph-based representations to model complex relationships and contexts [32, 31]. Knowledge Graphs play a crucial role in improving recommendation accuracy by organizing information into entities and relationships, thereby enriching the semantic understanding of user preferences. Integrating KGs with GCNs enables recommendation systems to incorporate rich contextual information and capture nuanced user-item interactions that traditional methods often overlook [58, 56].

Session-Based Recommendation Systems have gained prominence in scenarios where user preferences are volatile and short-lived, such as online shopping or content streaming platforms. Unlike traditional systems that rely on long-term user profiles, SBRS predict user preferences within individual sessions, making them adaptable to rapidly changing user behaviors [24, 45]. This research introduces a novel session-based next-item recommendation model that utilizes KGs to uncover user intents within sessions. Existing recommendation models often struggle with basic relational modeling, limiting their ability to capture detailed user-item relationships and long-term connections. Our proposed approach, inspired by hierarchical semantic networks, addresses these limitations by integrating KGs to better understand user intents and improve intra-session dependency modeling. By leveraging item auxiliary KGs, our model aims to provide more precise and personalized recommendations, improving recommendation accuracy and personalization while offering explainability for its suggestions [67, 25].

This thesis is structured as follows to systematically address these advancements and their applications:

Chapter 1 delves into the evolution and challenges of recommendation systems, with a focus on SBRS.

Chapter 2 provides a comprehensive overview of KGs and GCNs, elucidating their relevance and application in recommendation systems.

Chapter 3 outlines the methodology used in this research, detailing data preprocessing steps, KG integration, and GCN model architecture.

Chapter 4 presents experimental results and discussions, evaluating the effectiveness of the proposed approach.

In conclusion, this thesis contributes to advancing the field of recommendation systems by leveraging KGs and GCNs to address the complexities of dynamic user preferences. By enhancing recommendation accuracy and providing transparent explanations for recommendations, this research aims to empower users to navigate the vast digital landscape with confidence and satisfaction.

CHAPTER 1

Session Based Recommendation Systems

1 Introduction

Recommender Systems (RS) are essential for guiding users through large datasets by suggesting items of interest. Traditional approaches like collaborative filtering and content-based filtering, though effective, often struggle to capture dynamic and real-time user preferences. This limitation has led to the development of session-based recommendation systems, which focus on the sequence of user interactions within a single session. Session-based systems offer a timely and contextually relevant alternative, effectively capturing user preferences within specific contexts.
This chapter explores traditional recommendation systems and the motivation for session-based systems, highlighting their characteristics and challenges.

2 Recommendation Systems:

2.1 Definition:

Recommender Systems (RSs) are software tools and methods that offer recommendations for item[1]that could be useful to a user. The recommendations deal with several aspects of

[1]Is the general term used to denote what the system recommends to user

decision-making, like what to buy, what to listen to, and what news to read online[47]. By offering customers individualized, unique content and service recommendations, recommendation systems handle the issue of information overload that users typically face.[28] Recommendation systems are widely used in various domains, including:

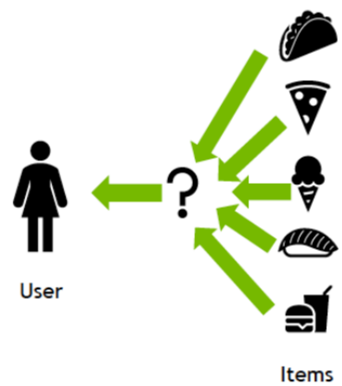


Figure 1.1: Recommender System Illustration
[5]

- **E-commerce**: Suggesting products to online shoppers.
- **Content Platforms**: Recommending articles, videos, music, or other media.
- **Social Networks**: Offering friend suggestions, group recommendations, etc.

2.2 TYPES OF RECOMMENDATION SYSTEMS:

The use of efficient and accurate recommendation techniques is very important for a system that will provide good and useful recommendation to its individual users. This explains the importance of understanding the features and potentials of different recommendation techniques.[28] While there are various recommendation algorithms and strategies, most can be categorized into the following:

- **Content-based:**Systems that utilize a content-based recommendation technique examine a collection of documents and/or user-rated item descriptions. Based on the characteristics of the objects the user has evaluated, the system creates a model or profile of the user's interests. The user's interests are arranged in their profile, which is used to suggest new and interesting items. In essence, the recommendation method compares a content object's attributes with the user profile's attributes.[35]

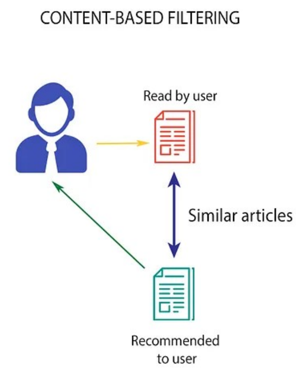


Figure 1.2: Content Based Recommendation System
[18]

- **Collaborative Filtering:**The most basic and original implementation of this approach involves recommending items to the active user that have been liked by other users with similar preferences. The similarity between users is determined by comparing their rating histories. Collaborative filtering is widely regarded as the most popular and extensively used technique in recommendation systems.[47]

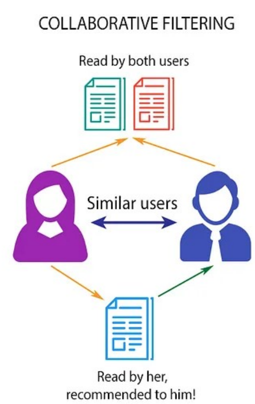


Figure 1.3: Collaborative Filtering Recommendation System
[18]

- **Hybrid recommendation system:** A hybrid recommendation system is a unique kind of recommender system that gives the user a recommendation by integrating two or more techniques, such as collaborative and content-based filtering. The problems with employing these two filtering techniques separately improved when they were combined.[39]

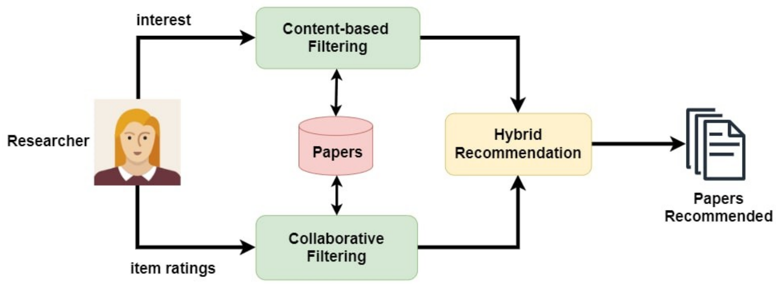


Figure 1.4: Hybrid Recommendation system
[3]

2.3 CHALLENGES AND LIMITATIONS:

1. **Cold Start:** The cold start problem occurs when the system is unable to form any relation between users and items for which it has insufficient data. There are two types of cold-start problems[33]:
   (a) **User Cold-Start Problem:** Occurs when there is limited information available about the user.
   (b) **Item Cold-Start Problem:** Arises when there is inadequate data about the product.
2. **Sparsity Problem:** The sparsity problem arises when users do not rate most items, leading to a sparse user-item matrix. This sparsity impacts the effectiveness of collaborative filtering methods, which rely on sufficient data to make accurate recommendations[53].
3. **Scalability:** Scalability is a challenge as recommender systems must handle increasing amounts of data efficiently. As the number of users and items grows, the computational demands increase, often leading to performance issues[53].

4. **Over-Specialization:** The over-specialization problem happens when users receive recommendations too similar to their existing preferences, limiting their exposure to new items. This restricts the diversity of recommendations, which is essential for a well-rounded user experience[53].

5. **Shilling Attack:** The Shilling Attack Problem arises when users falsify their identities to provide false ratings, compromising the system's reliability. Detecting and removing fake ratings are crucial steps in mitigating this issue[49].

6. **Synonymy:** The Synonymy Problem occurs when similar items have different representations, reducing recommendation accuracy. Strategies such as demographic filtering and term expansion are employed to address this challenge[49].

7. **Latency:** The Latency Problem in collaborative filtering emerges when new items are added, causing delays in recommendations. Employing offline calculations and clustering techniques can enhance performance and mitigate latency issues[49].

8. **Grey Sheep Problem:** The Grey Sheep Problem occurs in pure collaborative filtering approaches when a user's feedback does not match any user neighborhood, leading to inaccurate predictions. Utilizing pure content-based methods based on user profiles and item properties can help resolve this challenge[49].

9. **Evaluation and the Availability of Online Datasets:** Evaluating recommender systems is crucial for its effectiveness. However, selecting appropriate evaluation criteria and metrics poses a significant challenge. Traditional methods involve dividing datasets and using metrics like MAE, Precision, and F-Measure.may not be suitable for different domains. Alternative evaluation methods such as questionnaires, interviews, and user studies exist but can be costly and time-consuming. Another challenge is the limited availability of benchmark datasets tailored to specific domains, hindering accurate evaluation[30].

2.4 SESSION-BASED RECOMMENDATION SYSTEM

2.4.1 Motivation

Historically, the recommendation problem has often been approached as a matrix completion task. In this context, the objective is to predict preferences or ratings based on a set of user-item interactions, typically accumulated over extended periods. However, in many real-world applications, such long-term profiles are either unavailable or unusable because users are new, not logged in, or actively avoid tracking. These situations give rise to what is known as the session-based recommendation problem. The challenge here is to generate useful recommendations using only the limited information from the current session's recent user interactions [59].

2.4.2 Definition

Session-based recommendation systems focus on providing recommendations based solely on the interactions within the current user session. Unlike traditional systems that rely on long-term user profiles, session-based systems must infer user preferences from a very limited set of interactions, typically clicks, views, or purchases that occur during a single visit to a website or application [45].

2.4.3 Session and session properties

A session is a sequence of user interactions with a website or app within a short period. These interactions include actions like clicks, views, purchases, and searches. Each session has specific details, such as the items viewed, the time spent on each item, the order of item views, and the time gaps between actions.

Sessions usually show what a user currently prefers, but their intentions can change within the session[44]. In simple terms, a session is a list of interactions with a clear start and end,The system analyzes sequential data to generate real-time recommendations tailored to the user's immediate needs. Session-based recommender systems focus on predicting the unknown part of a session (see Figure **??**) or forecasting future sessions by modeling the complex relationships between multiple sessions.[59].

We will discuss five important properties of sessions that greatly affect session-based recommendation systems (SBRSs) [59]:

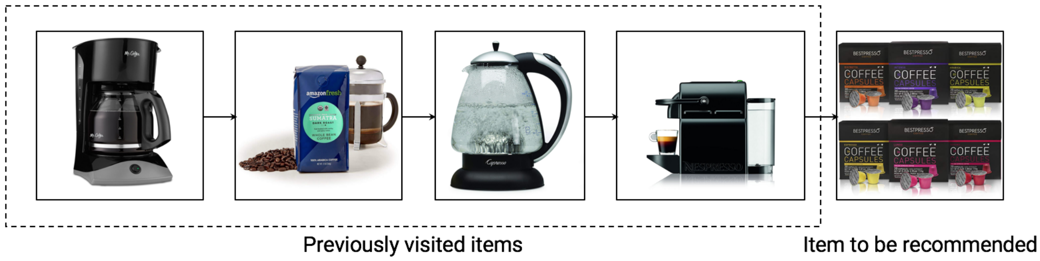


Figure 1.5: Hybrid Recommendation system

- **Property 1: Session Length:**

  Session length refers to the total number of interactions within a session. It serves as a basic yet crucial metric in session-based recommendation systems (SBRSs), often used as a statistical indicator in research literature[59].

- **Property 2: Internal Order:** Internal order in a session refers to the sequence of interactions within it. Sessions may display different levels of order flexibility, ranging from no specific order to a structured sequence. Internal order allows for the identification of sequential dependencies within sessions, which are valuable for making recommendations[59].

- **Property 3: Action Type:** Action type within a session refers to the variety of interactions occurring within it. Sessions may involve single actions, such as purchasing, or multiple actions, like clicking and purchasing. The nature of these actions determines the intra-session dependencies, which can be either homogeneous (based on a single type of actions) or heterogeneous (based on multi-type actions). This distinction is crucial for accurate recommendations[59].

- **Property 4: User Information:** User information within a session typically includes user IDs, and sometimes additional user attributes are also provided. In this context, this property refers to whether user information is present or absent in a session. The availability of user information is crucial for linking sessions from the same user across different time periods, enabling the modeling of long-term personalized preferences across multiple sessions. Session-based recommendation systems (SBRSs) were originally designed to handle sessions where user information is unavailable[59].

- **Property 5: Session-Data Structure:** Session-data structure refers to how session information is organized, typically with multiple levels. These levels may include details about individual interactions, user or item attributes, and historical sessions. The number of levels varies, impacting the amount of information available for recommendations[59].

2.4.4 SESSION-BASED RECOMMENDATION SYSTEMS CATEGORIZATION

According to [59],existing work on SBRSs can be classified into three sub-areas, providing a unified framework for categorization:

Next Interaction Recommendation:

- **Objective:** Recommends the next possible interaction within the ongoing session.
- **Focus:** Primarily models intra-session dependencies.
- **Example:** Predicting the subsequent item a user is likely to interact with during their current session.

Next Partial-Session Recommendation:

- **Objective:** Recommends all remaining interactions needed to complete the current session.
- **Focus:** Mainly models intersession dependencies.
- **Example:** Predicting additional items to complete a shopping basket based on items already purchased.

Next Session Recommendation:

- **Objective:** Recommends the content of the next session.
- **Focus:** Mainly models inter-session dependencies.
- **Example:** Recommending the content of the next basket based on past session data.

The Table 1.1 illustrates the differences between sub-areas in SBRSs.

Table 1.1: A comparison of different sub-areas in SBRSs

[59]

Sub-area	Input	Output & Typical research topic
Next interaction recommendation	Mainly known part of the current session	Next interaction (item). Typical research topics: Next item recommendation, next song/movie recommendation, next POI recommendation, next web page recommendation, next news recommendation, etc.
Next partial-session recommendation	Mainly known part of the current session	Subsequent part of the session. Typical research topics: Next items recommendation, session/basket completion
Next session recommendation	Historical sessions	Next session. Typical research topics: Next basket recommendation, next bundle recommendation, etc.

2.4.5 Comparison of SBRS with Traditional RS Approaches

Table 1.2 presents a comprehensive comparison between SBRS and other typical RSs.

	SBRS	CF	CB	Hybrid Recommender Systems
Data Source	User session interactions	User-item interactions	Item features	Multiple sources (session, CF, CB)
Recommendation Type	Session-based recommendations	User-based or item-based recommendations	Item-based recommendations	Combined recommendations
Personalization	Highly personalized	Moderately personalized	Moderately personalized	Highly personalized
User Context	Captures temporal context	Temporal context limited	No explicit context captured	Can exploit multiple types of context
Sequential Patterns	Exploits sequential behavior	Doesn't consider sequential patterns	Doesn't consider sequential patterns	Can exploit sequential patterns
Cold Start Problem	Suffers from cold start problem	Suffers from cold start problem	Less affected by cold start problem	Suffers from cold start problem
Scalability	Can face scalability challenges	Scalable	Scalable	Scalable
Sparsity	Handles sparse data effectively	Requires denser data for accuracy	Handles sparse data effectively	Handles sparse data effectively
Serendipity	Can offer serendipitous recommendations	Moderate serendipity	Limited serendipity	Moderate serendipity
Explanation	Limited explicit explanations	Lacks explicit explanations	May provide feature-based explanations	May provide combined explanations

Table 1.2: Comparison of SBRS with Traditional RS Approaches
[57]

3 Characteristics and Challenges

According to [55], gaining a comprehensive understanding of the characteristics of session data and the challenges associated with modeling it is crucial in order to develop a well-suited Session-Based Recommender System (SBRS). In this section, we illustrate and summarize these characteristics and challenges as follows:

- **Related to Session Length**
  - **Long Sessions:** A session that is considered long typically consists of a higher number of interactions, exceeding 10 or more. The challenges include how to effectively reduce noisy information from irrelevant interactions and how to learn complex dependencies for better recommendation performance.
  - **Medium Sessions:** Sessions of medium duration encompass a moderate number of interchanges, ranging from approximately 4 to 9 interactions. The challenge lies in developing SBRSs for medium length sessions, which remains fundamentally difficult.
  - **Short Sessions:** These sessions consist of limited interactions, usually less than 4, which limits the information available for recommendations.
- **Related to Internal Order**
  - **Unordered Sessions:** These are sessions without any chronological order between interactions. The challenges involve learning co-occurrence-based dependencies and capturing collective dependencies[59].
  - **Ordered Sessions:** These sessions have a strict order among interactions. The challenge is to effectively learn cascaded long-term sequential dependencies[59].
  - **Flexibly-Ordered Sessions:** These sessions have a mixed order. The challenges are learning complex and mixed dependencies[59].
- **Related to Action Type**
  - **Single-Type-Action Sessions:** These sessions include only one type of action. The challenge is to learn dependencies from the same type of actions[59].

  - **Multi-Type-Action Sessions:** These sessions include more than one type of action. The challenges include learning complex dependencies across different types of actions[59].

- **Related to User Information**
  - **Non-Anonymous Sessions:** These are sessions with associated user information. The challenge is to precisely learn personalized long-term preferences[59].
  - **Anonymous Sessions:** These are sessions without associated user information. The challenge is capturing personalized preferences with limited contextual information[59].

- **Related to Session-Data Structure**
  - **Single-Level Session Data:** This data contains only inter-interaction dependencies. The challenges include overcoming cold-start and sparsity issues[59].
  - **Multi-Level Session Data:** This data contains intra- and inter-level dependencies. The challenge is to learn dependencies within and across different levels[59].

4 TAXONOMY OF SBRS APPROACHES

Session-Based Recommender Systems (SBRS) utilize various methodologies to provide recommendations. These approaches can be broadly categorized into three main types: Conventional SBRS Approaches, Latent Representation-Based Approaches, and Deep Neural Network-Based Approaches. Here we provide an overview of these categories, with a particular focus on K-nearest neighbor (KNN) and Graph Neural Networks (GNN) approaches.

4.1 CONVENTIONAL SBRS APPROACHES

Conventional SBRS approaches employ traditional algorithms and techniques to analyze session data and make recommendations[59]. These include:

- **Pattern/Rule Mining**: Techniques that identify frequent patterns or rules in session data[59].

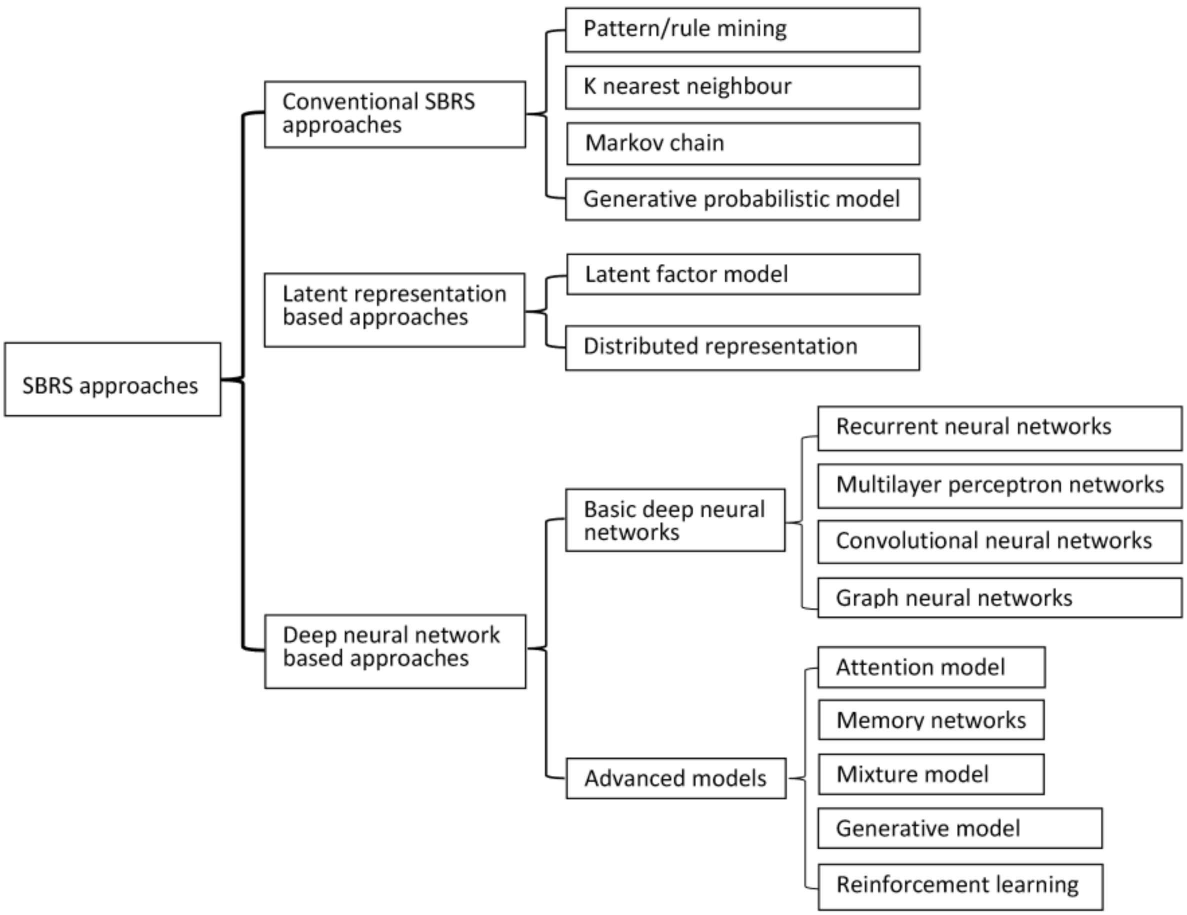


Figure 1.6: Categorization of SBRS approaches

- **K-Nearest Neighbor (KNN)**: This approach recommends items by finding similar sessions based on the distance metric[59]. We will discuss this approach in more detail later.
- **Markov Chain**: Utilizes probabilistic models to predict the next item based on the previous items in a session[59].
- **Generative Probabilistic Model**: Models that generate probable next items based on learned probability distributions[59].

4.2 Latent Representation-Based Approaches

These approaches focus on transforming session data into latent spaces where similar items or sessions are closely positioned:

- **Latent Factor Model**: Uses matrix factorization techniques to represent items and sessions in latent spaces[59].
- **Distributed Representation**: Embeds items and sessions into continuous vector spaces using techniques like word2vec[59].

4.3 Deep Neural Network-Based Approaches

Deep learning models have been increasingly applied to SBRS, leveraging their ability to capture complex patterns in data. These approaches include:

- **Basic Deep Neural Networks**:
  - **Recurrent Neural Networks (RNN)**: Captures sequential dependencies in session data[59].
  - **Multilayer Perceptron Networks (MLP)**: Fully connected networks that learn non-linear interactions[59].
  - **Convolutional Neural Networks (CNN)**: Extracts spatial hierarchies in session data[59].
- **Graph Neural Networks (GNN)**: Models the relationships between items as a graph. This approach will be discussed in more detail[59].

- **Advanced Models**:
  - **Attention Model**: Focuses on important parts of the session data[59].
  - **Memory Networks**: Utilizes external memory for capturing long-term dependencies.
  - **Mixture Model**: Combines several probabilistic models to improve recommendation[59].
  - **Generative Model**: Generates recommendations by modeling the data distribution[59].
  - **Reinforcement Learning**: Uses a reward-based system to optimize recommendations[59].

4.4 K NEAREST NEIGHBOUR BASED SBRSS

Now, let's delve into K Nearest Neighbour (KNN) based approaches for SBRS. These approaches have demonstrated simplicity and effectiveness. In essence, a KNN-based SBRS identifies the K interactions or sessions most similar to the current interaction or session, respectively, from the session data. Subsequently, it computes a score for each candidate interaction based on its similarity to the current interaction, serving as guidance for recommendations. For consistency with the previous discussion, each interaction is treated as an item in this class of approaches. Based on whether the similarity is calculated between items or sessions, KNN-based approaches for SBRSs can be categorized into item-KNN and session-KNN[59].

4.4.1 ITEM-KNN

In an item-KNN based SBRS, given the current session context, recommendations are made for the K items most similar to the current item in terms of their co-occurrence in other sessions. Technically, each item is encoded into a binary vector where each element indicates whether the item occurs (set to "1") in a specific session or not (set to "0"). Consequently, the similarity between items can be calculated on their vectors using a similarity measure such as cosine similarity[59].

4.4.2 SESSION-KNN

In a session-KNN based SBRS, given the current session context c, the system first computes the similarity between c and all other sessions to identify the set $N(c)$ of its K neighbor sessions. It then calculates the score of each candidate item $\hat{v}$ with respect to c based on this similarity:

$$\text{score}(\hat{v}) = \sum_{s_{nb} \in N(c)} \text{sim}(c, s_{nb}) \cdot 1_{s_{nb}}(\hat{v})$$

where sim represents a similarity measure and $1_{s_{nb}}(\hat{v})$ is an indicator function that returns 1 if $\hat{v}$ occurs in s_{nb} and 0 otherwise[59].

Compared to item-KNN, session-KNN considers the entire session context rather than just the current item, enabling it to capture more information for more accurate recommendations[59].

4.5 Graph Neural Networks (GNN) based SBRSs

Graph Neural Networks (GNNs) have shown significant power in modeling complex relations in graph-structured data. By integrating deep neural networks into graph data, GNNs can model complex transitions within or between sessions to improve SBRS performance. The process involves transforming session data into a graph, where each session becomes a chain, each interaction is a node, and edges connect adjacent interactions. This graph is processed by a GNN to learn embeddings for each node, which are used for session-based recommendations. GNN approaches in SBRSs are generally divided into three classes: Gated Graph Neural Networks (GGNN), Graph Convolutional Networks (GCN), and Graph Attention Networks (GAT)[59].

4.5.1 Gated Graph Neural Networks (GGNN) for SBRSs

In GGNN-based SBRSs, a directed graph is constructed from historical sessions, with edge directions indicating the order of interactions. Each session graph is processed by GGNN to obtain node embeddings. A Gated Recurrent Unit (GRU) updates each node's embedding recurrently, considering its previous hidden state and the states of its neighbors:

$$\mathbf{h}_i^{(t)} = \text{GRU}\left(\mathbf{h}_i^{(t-1)}, \sum_{n_j \in N(n_i)} \mathbf{h}_j^{(t-1)}, \mathbf{A}\right)$$

where $N(n_i)$ is the set of neighboring nodes of n_i, and $\mathbf{A}$ is the adjacency matrix. The final hidden state of node n_i after multiple iterations is taken as its embedding. GGNNs have demonstrated superior performance compared to non-GNN approaches[59].

4.5.2 Graph Convolutional Networks (GCN) for SBRSs

GCN-based SBRSs use pooling operations to integrate information from a node's neighbors, helping update its hidden state:

$$\hat{\mathbf{h}}_i^{(t)} = \text{pooling}\left(\{\mathbf{h}_j^{(t-1)}, n_j \in N(n_i)\}\right)$$

Different pooling methods, like mean or max pooling, can be used. The neighborhood information is then incorporated into the node's hidden state update:

$$\mathbf{h}_i^{(t)} = \mathbf{h}_i^{(t-1)} + \hat{\mathbf{h}}_i^{(t)}$$

The final hidden state of each node, when a stable equilibrium is reached, is taken as its embedding[59].

4.5.3 Graph Attention Networks (GAT) for SBRSs

GAT-based SBRSs use an attention mechanism to integrate information from a node's neighbors in a session graph:

$$\mathbf{h}_i^{(t)} = \text{attention}\left(\{\mathbf{h}_j^{(t-1)}, n_j \in N(n_i)\}\right)$$

where $\mathbf{h}_i^{(t)}$ is the hidden state of node n_i at the t-th attention layer. The attention mechanism calculates the importance weights of each neighboring node and aggregates their hidden states. The final hidden state of each node after multiple attention layers is taken as its embedding. GAT approaches like Full Graph Neural Network (FGNN) effectively capture item transition patterns within sessions[59].

5 Conclusion

In this chapter, we provided an overview of traditional and session-based recommender systems, highlighting their respective strengths and motivations. We underscored the need for session-based approaches in dynamic user environments.

In the next chapter, we will explore the use of Knowledge Graphs (KG) and Graph Convolutional Networks (GCN) to enhance session-based recommendations, along with a review of related work to lay the groundwork for our proposed methodology.

CHAPTER 2

Background

1 Introduction

Knowledge graphs (KGs) play a crucial role in modern recommender systems, enhancing recommendation accuracy through structured data. This chapter explores their integration with Graph Neural Networks (GNNs) in session-based recommendation systems. It covers an overview of KGs, including definition, types, and applications, followed by an exploration of GNNs, focusing on fundamental concepts and applications such as Graph Convolutional Networks (GCNs). The chapter also reviews recent advancements in session-based recommendation systems that combine KGs and GNNs.

2 Knowledge Graph

2.1 Definition

- **Heterogeneous Information Network:**A Heterogeneous Information Network (HIN) is a directed graph $G = (V, E)$ with an entity type mapping function $\phi : V \to A$ and a link type mapping function $\psi : E \to R$. Each entity $v \in V$ belongs to an entity type $\phi(v) \in A$, and each link $e \in E$ belongs to a relation type $\psi(e) \in R$. In addition, the number of entity types $|A| > 1$ or the number of relation types $|R| > 1$.[23]

- **Knowledge Graph:**A knowledge graph (KG) is a structured form of data that describes real-world entities—such as objects, events, situations or concepts—[27] and their relationships. It defines possible classes and relations of entities within a schema, enabling the potential interrelation of arbitrary entities across various topical domains[19].

  Typically, a KG is represented as a directed graph $G = (V, E)$, where V denotes the vertices representing real-world entities, and E represents the edges that indicate relationships between these entities. These vertices (nodes) are interconnected by edges, which are the relations in the graph[8].A KG can be considered as an instance of a HIN[23].

2.2 Example of a Knowledge Graph

To illustrate the concept of a Knowledge Graph, consider the example depicted in Figure 2.1. This graph represents a subset of the relationships involving Tim Berners-Lee, the inventor of the World Wide Web (WWW).

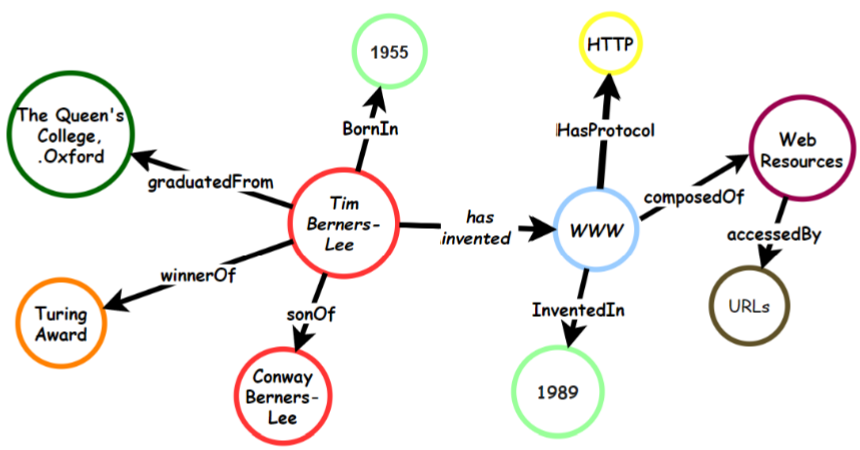


Figure 2.1: An example of a Knowledge Graph
[8]

- The vertices (nodes) represent entities such as "Tim Berners-Lee," "WWW," "The Queen's College, Oxford," and "Turing Award."

- The edges represent relationships between these entities, such as:

  - "invented" between Tim Berners-Lee and WWW.
  - "graduated from" between Tim Berners-Lee and The Queen's College, Oxford.
  - "winner of" between Tim Berners-Lee and Turing Award.

- The graph shows:
  - Tim Berners-Lee was born in 1955.
  - He invented the WWW in 1989.
  - The WWW uses the HTTP protocol.
  - Web resources are accessed by URLs.

- This example demonstrates how KGs model complex relationships in a structured way.

2.3 Structured Data Representation in Knowledge Graphs

Structured data plays a fundamental role in knowledge graphs, providing a predefined organization of information that enables clear definition and understanding of relationships between different entities. Typically organized in rows and columns, structured data allows for easy search and analysis, facilitating better data integration, interoperability, and analytics within knowledge graphs.

Several significant large-scale open knowledge graphs act as crucial resources for organizing and making accessible extensive structured information, including:

2.3.1 DBpedia

DBpedia is a community-driven project that extracts structured content from the information created as part of the Wikipedia project. It allows users to query relationships and properties associated with Wikipedia resources, including links to other related datasets. DBpedia is significant because it converts Wikipedia content into structured data, making it accessible and usable in a variety of semantic web and linked data applications[4].

2.3.2 FREEBASE

Freebase was a large collaborative knowledge base consisting of data composed mainly by its community members. It provided a structured database of information collected from many sources, including individual, user-submitted data. Freebase's significance lies in its contribution to the development of Google's Knowledge Graph, enhancing the search engine's ability to understand and interpret user queries more contextually[11].

2.3.3 WIKIDATA

Wikidata is a free and open knowledge base that can be read and edited by both humans and machines. It acts as a central storage for the structured data of Wikimedia projects, including Wikipedia. Wikidata is significant because it supports a wide range of applications, from infoboxes in Wikipedia articles to more complex queries and data integrations, thanks to its structured and linked data format[6].

2.4 TYPES OF KNOWLEDGE GRAPH

- **Item Knowledge Graph:**Items and item-associated entities, such as item attributes, function as nodes in the item KG. Edges may represent the attribute level of an item. connections, like brand, category, or user-related connections, like "co-view" and "co-buy."[23]

- **User-Item Knowledge Graph:** Users, items, and the entities they are linked with are the nodes in the user-item KG. In addition to the relationships between the item and the user that are contained in the item KG, the user-item KG also includes relationships like "buy," "click," and "mention."[23]

2.5 HISTORICAL BACKGROUND

The concept of knowledge graphs (KGs) has developed significantly over time. The earliest KGs were created as semantic networks for a university project from 1972 to 1980. Between 1985 and 2007, the idea of KGs was applied to various fields, such as human language in projects like WordNet[1]. During this period, algorithms focused on using the symbolic semantics of KGs to reason and learn new rules.[52]

The term "knowledge graph" was first used informally in the 1980s by academics from the Universities of Groningen and Twente in the Netherlands. They described a knowledge-based system that integrated knowledge from various sources to represent natural language. Their idea of KGs, which had a limited number of relations and emphasized qualitative modeling with human interaction, is quite different from today's more complex KGs.

The success of KGs grew rapidly with the creation of general-purpose KGs like DBpedia[2] and Freebase[3], leading to Google's Knowledge Graph in 2012. Google popularized the term "Knowledge Graph" through its blog post "Introducing the Knowledge Graph: things, not strings"[4] which introduced it as a way to search for real-world objects rather than just strings of text. Despite the blog post's lack of technical details, it has been cited more than 100 times according to Google Scholar since 2012. Since then, many private companies and academic institutions have used KGs for various applications.[19]

2.6 KNOWLEDGE GRAPH APPLICATIONS

In recent years, knowledge graphs have become a popular means for modeling relational data[37], adopted in various industrial and academic applications,the Figure 2.2 shows several applications of KGs

- **Question Answering System:** Semantic data derived from knowledge graphs can enrich search outcomes in semantic-aware question answering (QA) services(**eg:** Watson, a QA system developed by IBM, utilizes multiple knowledge bases like YAGO and DBpedia as its primary data sources. These knowledge bases enable Watson to effectively compete against human experts, showcasing the value of knowledge graphs in powering QA systems). Furthermore, structured knowledge assumes a crucial role social chatbots and digital assistants (**eg:**XiaoIce[5], Cortana[6] and Siri[7])[69]

[1] https://wordnet.princeton.edu/
[2] https://www.dbpedia.org/
[3] https://developers.google.com/freebase?hl=fr
[4] https://blog.google/products/search/introducing-knowledge-graph-things-not/
[5] https://www.xiaoice.com/
[6] https://www.microsoft.com/en-us/cortana
[7] https://www.apple.com/fr/siri/

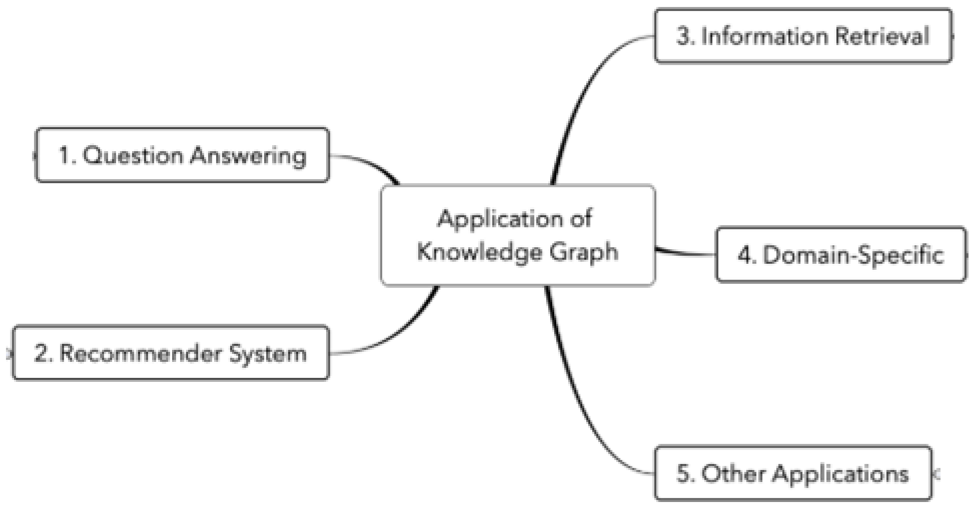


Figure 2.2: Application fields of KGs

- **Recommender System:** Recent research has started to investigate knowledge graphs (KGs) as an additional source of information to enhance recommender systems. The relationships embedded within KGs contribute to improving the accuracy of recommendations and increasing the variety of suggested items. Additionally, KGs offer interpretability to recommender systems[69]. For example [61]proposed a method for learning the intents behind user interactions with knowledge graphs to improve recommendation performance.

- **information retrieval:** Knowledge graphs (KGs) enhance information retrieval (IR) systems by offering structured knowledge about real-world entities. They enrich queries by incorporating related entities and textual information, improving query representation. For example, the work [17]demonstrates how features from entities themselves and links between entities to knowledge bases, such as structured attributes and text, can be utilized to enrich queries. Additionally, KGs contribute to advanced ranking models by establishing connections between queries and documents through related entities, resulting in more accurate and relevant search results[69].

- **Domain-Specific:** knowledge graphs are increasingly used for their capacity to manage relational data effectively[69], we present some key applications of KGs within specic domain:

- **Medical:** KGs enhance the retrieval and integration of medical knowledge. Example:[54].
- **Cyber Security:** KGs improve threat detection and prediction in cybersecurity. Example:[29].
- **Finance:** KGs assist in stock price prediction and financial analysis. Example:[34].
- **News:** KGs support news recommendation and fake news detection. Examples:[58, 16].
- **Education:** KGs enable learning resource recommendation and concept visualization. Examples:[15, 20].

- **Other Applications:** KGs have diverse applications, aiding in social network de-anonymization and privacy inferring (**eg:** [43]), classification tasks such as image classification (**eg:** [65]) and sentiment analysis (**eg:**[36]), knowledge extraction from geological documents (**eg:**[68]), and facilitating machine translation through multilingual knowledge graph embeddings (**eg:**[38, 14]).

2.7 KNOWLEDGE GRAPH EMBEDDING

Knowledge Graph Embedding (KGE) involves transforming a knowledge graph $G_{\text{know}} = (V, E)$ into a lower-dimensional space. Following the embedding process, each component of the graph, such as entities and relations, is represented as a vector in a d-dimensional space. Despite the reduction in dimensionality, the embedding retains the essential characteristics of the graph, allowing for the quantification of semantic meaning or high-order proximity within the graph [22].

3 GRAPH NEURAL NETWORK

3.1 OVERVIEW

Graphs are fundamental structures employed to model complex relationships among entities. Their versatility has led to their widespread application in various fields, including social network analysis, recommendation systems, computer networks, and bioinformatics. In recent years, the advent of deep learning techniques has extended to graphs, facilitating the extraction of features and learning of representations through Graph Neural Networks (GNNs). GNNs have demonstrated significant potential across numerous applications, particularly in session-based recommender systems.

In this section, we present an introduction to graphs and their critical role in machine learning. We explore the various types of graphs, and examine different graph representations. Additionally, we provide a concise overview of GNNs, highlighting their relevance and effectiveness in the context of session-based recommender systems.

3.2 Main Concepts

1. **Graph**: A graph is defined as an ordered pair $G = (V, E)$, where V represents a finite nonempty set of vertices, and E denotes a set of 2-element subsets of V, referred to as edges. Vertices are also known as points or nodes, while edges are referred to as lines or links. When E consists of ordered pairs of distinct vertices, the graph G is termed a Directed Graph or DiGraph[42].

2. **Size and Order of a Graph**: The size of a graph $G = (V, E)$ is quantified by the number of edges $|E|$, while the order is determined by the number of vertices $|V|$. A graph of trivial order consists of a single vertex and no edges, thus having an order of one and a size of zero[42].

3. **Adjacency and Incidence**: In graph G, a pair of vertices (u, v) are adjacent or neighbors if there exists at least one edge connecting them. An edge is considered incident on a vertex v if it either originates or terminates at v[42].

4. **Degree of a Vertex**: The degree of a vertex v in graph G, denoted as deg(v), is the count of edges incident on v or, equivalently, the number of vertices adjacent to v. In a directed graph, the number of edges directed into and out of vertex v are referred to as the indegree and outdegree, respectively[42].

5. **Neighborhood**: The neighborhood $N(v)$ of a vertex v in graph G is defined as the set of vertices adjacent to v. The neighborhood graph of v is an induced subgraph consisting of all vertices in $N(v)$ and the edges connecting these vertices[42].

6. **Adjacency Matrix:** A graph $G(V, E)$ of order n, where $|V| = n$ and size m, where $|E| = m$, is represented by its adjacency matrix $A = [a_{ij}]$, which is an $n \times n$ matrix defined as follows[42]:

$$a_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E \\ 0, & \text{otherwise} \end{cases}$$

7. **Degree Matrix:**Given a graph $G = (V, E)$ with $|V| = n$, the degree matrix D for G is an $n \times n$ diagonal matrix defined as follows:

$$D_{i,j} = \begin{cases} \deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

   The degree $\deg(v_i)$ of a vertex v_i is the number of edges incident to that vertex. In an undirected graph, a loop contributes two to the degree of a vertex. In a directed graph, the degree can be specified as either the indegree (the number of incoming edges to a vertex) or the outdegree (the number of outgoing edges from a vertex)[57].

8. **Laplacian Matrix**

   Given a simple graph G with n vertices $v_1, \ldots, v_n$, the Laplacian matrix $L_{n\times n}$ is defined element-wise as:

$$L_{i,j} = \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

   Alternatively, the Laplacian matrix can be expressed as:

$$L = D - A$$

   where D is the degree matrix and A is the adjacency matrix of the graph. Since G is a simple graph, A contains only 1s or 0s, and its diagonal elements are all 0s[57].

9. **Message Passing in Graph Neural Networks:** Feature information associated with a graph often includes node-level attributes and edge-level attributes. Neural Message Passing (NMP) is a framework in which node features (vector messages) are exchanged between nodes to determine hidden embeddings, which are the representation vectors corresponding to each node. A hidden embedding $h_v^{(k)}$ is obtained from the aggregated information from the neighborhood $N(v)$ of node v at time step k. The NMP approach, also known as the message passing update, involves a two-step procedure: the first step is to aggregate messages, and the next is to update node embeddings. The NMP update process leverages the graph structure that can be obtained from the adjacency matrix, and node features are indicated as X. Starting with initializing node embeddings $h^{(0)} = X$, the two steps are performed in each layer of the model [7].

   - **Aggregate:** The aggregated embedding for a node is obtained by collecting the feature vectors of all immediate neighbor nodes.
   - **Update:** The new embedding vector for the node is updated by considering the existing feature of the node and the aggregated representation vector from neighbor nodes.

     The equations below show a step-by-step mathematical depiction of the two stages, which can be defined as the general framework of a GNN model.

Algorithm 1 Message Passing Steps [7]

1: Initialize: $h^{(0)} = X$
2: **for** $k = 1, 2, ..., K$ **do**
3: $\quad agg^{(k)} = \text{Aggregate}^{(k)}\{h^{(k-1)} : u \in N(v)\}$
4: $\quad h^{(k)} = \text{Update}^{(k)}\{h^{(k-1)}, agg^{(k)}\}$
5: **end for**

Here, $\text{agg}_v^{(k)}$ is the aggregated information for node v from its neighborhood $N(v)$. $N(v)$ contains all the vertices that share an edge with v. A vertex u is known as a neighbor of v if there exists an edge e_{uv} or a path of length k between the nodes u and v. When $k = 0$, each node v is initialized with its feature vector x_v. With each increment of k, $k = 1$, the updated embedding considers the representation vector of neighbor nodes with path length one, and similarly K-hop neighbors for $k = K$. The k-th update $h_v^{(k)}$ for node v and layer k is obtained by considering the aggregated message $\text{agg}_v^{(k)}$ and the feature embedding from the previous layer $k-1$ for node v, $h_u^{(k-1)}$.

The aforementioned Aggregate and Update methods satisfy the fundamental requirements for designing neural networks since they are differentiable functions. Therefore, an activation function can be applied to these methods to train the model, and most importantly, backpropagation and forward pass can be performed without hindrance. The Figure 2.3 illustrates the message aggregation process implemented by

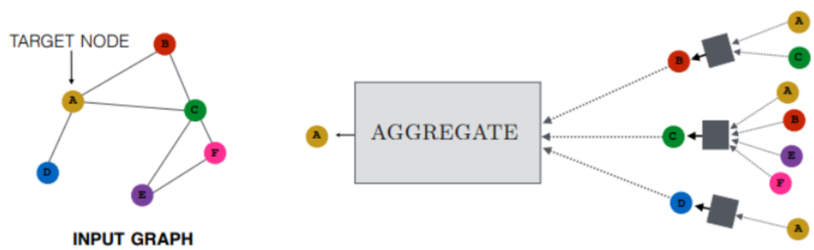


Figure 2.3: Neural Message Psassing
[7]

a single node in its neighborhood. The image depicts a two-layer message-passing framework. The embedding for node A is associated with the feature vectors of $N(A) = \{B, C, D\}$, subsequently aggregating the feature vectors of each node in $N(A)$ from its neighbor node embeddings. Therefore, the node embedding update for node v and $k = 2$ depends on the feature vector of nodes in $N(A) = \{B, C, D\} = N(N(B), N(C), N(D)) = \{B, \{A, C\}, C, \{A, B, E, F\}, D, \{A\}\}$. The message-passing approach in GNN unrolls the graph structure into a tree of depth k by unraveling the neighbor nodes of the selected node.[7]

3.3 GNN Tasks

According to [2], Graph Neural Networks (GNNs) cover diverse tasks tailored to address specific challenges in graph-structured data. These tasks include:

- **Node Classification:** This task utilizes neighboring node labels to predict missing node labels within a graph.

- **Link Prediction:** It predicts the link between a pair of nodes in a graph with an incomplete adjacency matrix. This task finds common application in social networks.
- **Community Detection:** This task involves dividing nodes into various clusters based on edge structure. It learns from edge weights, distances, and graph objects.
- **Graph Embedding:** It maps graphs into vectors, preserving relevant information on nodes, edges, and structure.
- **Graph Generation:** This task learns from sample graph distributions to generate new but similar graph structures.

The Figure 2.4 illustrates GNN tasks.

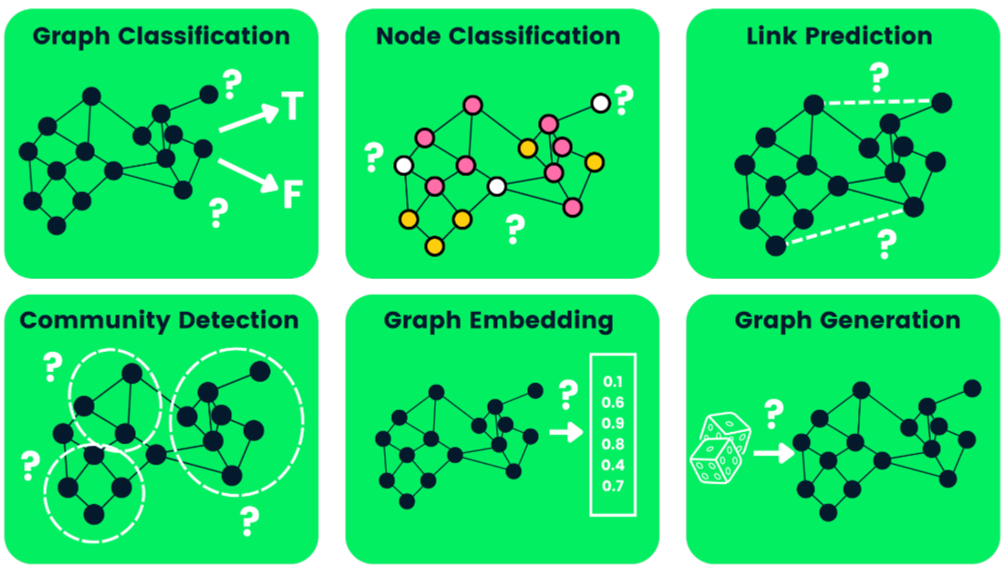


Figure 2.4: Graph Neural Network tasks
[2]

3.4 Types of Graph Neural Networks

3.4.1 Graph Convolutional Networks (GCNs)

A GCN is a type of GNN that utilizes convolutional layers to process graph data. These layers apply a set of learnable filters to the graph, designed to consider the structure of the graph and the relationships between vertices[57].

3.4.2 Graph Autoencoders (GAEs)

GAEs employ graph convolutional layers to acquire a low-dimensional representation of the input graph. The network is trained to encode the graph into a lower dimensional space and decode it back to the original graph[57].

3.4.3 Graph Recurrent Networks (GRNs)

GRNs are tailored for processing graph-structured data in a sequence or time-series setting. They leverage recurrent neural networks to propagate information between nodes across several time-steps[57].

3.4.4 Graph Transformers

Graph Transformers draw inspiration from the Transformer architecture utilized in natural language processing tasks. They utilize self-attention mechanisms to capture the relationships between nodes in the graph[57].

3.4.5 Graph Attention Networks (GATs)

GATs, a type of GNN, employ an attention mechanism to weigh the importance of different vertices in a graph when processing data. This enables GNNs to concentrate on the most relevant elements and relationships when making predictions[57].

3.5 GNNs advantages and limitations

Graph neural networks (GNNs) present notable advantages, particularly their capability to process and analyze complex graph-structured data. Their versatility is demonstrated by their applicability to both supervised and unsupervised learning tasks. Despite these strengths, GNNs do have certain limitations. One significant drawback is their computational expense, which can become particularly pronounced with large graphs. Furthermore, GNNs are susceptible to overfitting, especially in the presence of noisy or incomplete graph structures. Lastly, the interpretability of GNNs remains a substantial challenge,as understanding the processes by which these networks generate predictions can often be challenging[57].

3.6 Graph Convolutional Networks

3.6.1 Main Idea

As implied by the term "Convolutional," the concept originated from image processing and was subsequently adapted to graphs. While images have a fixed structure, graphs are significantly more complex.[57]

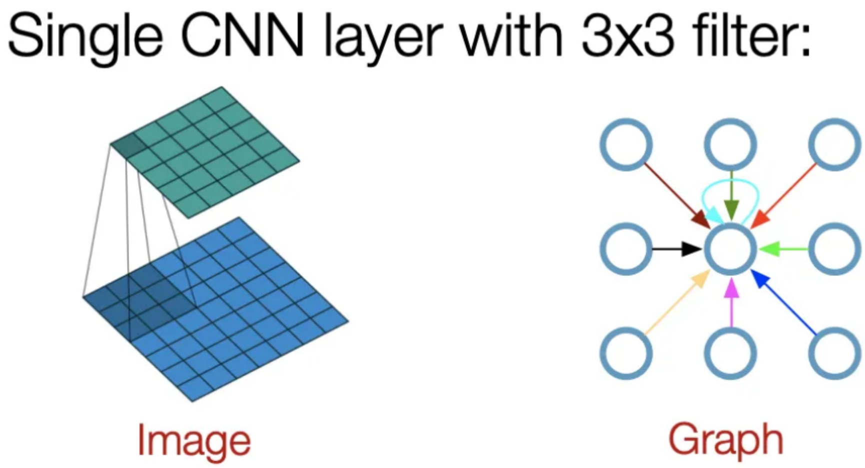


Figure 2.5: Convolution idea from images to graphs
[18]

3.6.2 Definition and Principles

Graph Convolutional Networks (GCNs) are a class of deep learning models specifically designed to operate on graph-structured data. Unlike traditional neural networks that process grid-like data, GCNs extend neural network architectures to handle non-Euclidean domains represented by graphs or networks. The core idea behind GCNs is to generalize the concept of convolutional layers from grid-like data, such as images, to graphs. In traditional Convolutional Neural Networks (CNNs), convolutions operate on local neighborhoods of pixels, leveraging the grid structure. In contrast, GCNs define convolutions in the spectral or spatial domain of graphs, utilizing the connectivity patterns between nodes[57]..

GCNs typically operate within a message-passing framework, where each node receives and aggregates information from its neighboring nodes. This aggregation process is analogous to the receptive field in CNNs, allowing nodes to gather information from their local graph neighborhood. The gathered information is then used to update the node's representation or features. By iteratively propagating and aggregating information across the graph, GCNs learn to capture the graph structure and perform node-level or graph-level predictions.[57]

3.6.3 GCN ARCHITECTURE

Graph Convolutional Networks (GCNs) can be intrpreted as analogous to convolutional filters in traditional neural networks, adapted to operate on irregular structures like graphs rather than regular grids, as depicted in Figure 2.6. The node embeddings in GCNs are updated iteratively according to the formula[57].:

$$\mathbf{H}^{(l+1)} = \delta\left(\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

Here, $\mathbf{H}^{(l)}$ represents the embedding matrix of the l-th convolutional layer, $\mathbf{D}$ is the degree matrix of the adjacency matrix $\mathbf{A}$ which includes self-loops, and $\mathbf{W}^{(l)}$ are the trainable weights in the GCN layer. This formulation ensures that each node aggregates information from its neighbors across multiple hops in the graph[57]..

GCNs are simplified versions of Graph Convolutional Neural Networks (GCNNs), typically comprising three main steps[57].:

1. Feature Propagation
2. Linear Transformation
3. Application of Non-linear Activation function

- **Feature Propagation (Convolution and Message Passing):**

  In the feature propagation step, GCNs aim to capture and propagate information across the graph by considering the features of neighboring nodes. This is achieved by computing a weighted sum of the features of each node's neighbors and incorporating it into the node's own feature representation. The weights are typically determined based on the graph structure or learned through a training process. This aggregation

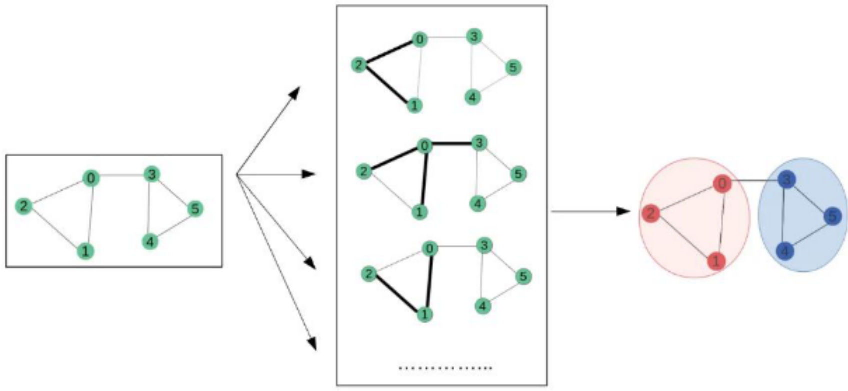


Figure 2.6: Illustration of Graph Convolutional Networks
[18]

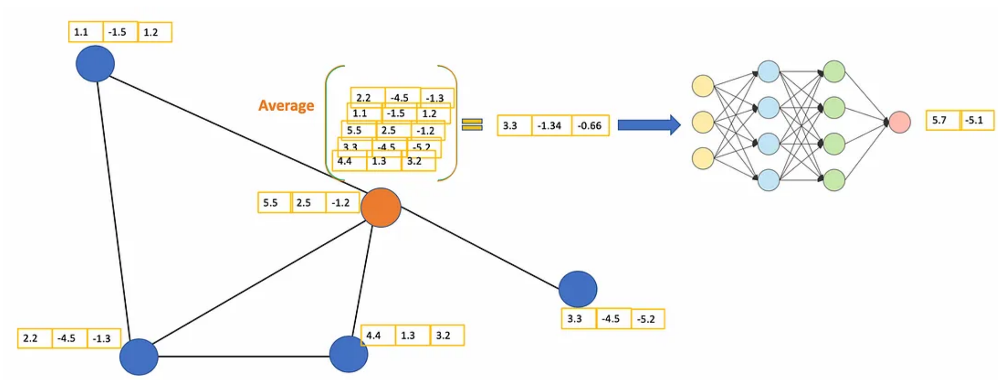


Figure 2.7: Illustration of feature propagation in GCN (orange node).

of neighboring features helps to capture the local context and dependencies in the graph [57].

Consider Figure 2.7, where for each node, we gather the feature information from all its neighbors, including itself. Assume we use the average function for this aggregation. This process is repeated for all nodes, and the averaged values are subsequently fed into a neural network. The central idea of GCNs is illustrated using the orange node in Figure 2.7. First, the average of all its neighbors, including itself, is computed. This average value is then passed through a neural network, which in the case of GCNs, is typically a fully connected layer. In this example, we obtain 2-dimensional vectors as the output, corresponding to the two nodes in the fully connected layer.

An important aspect to consider is demonstrated in Figure 2.7, which illustrates an example of a 2-layer GCN. The output of the first layer serves as the input for the second layer. Notably, the neural network in a GCN is simply a fully connected layer .

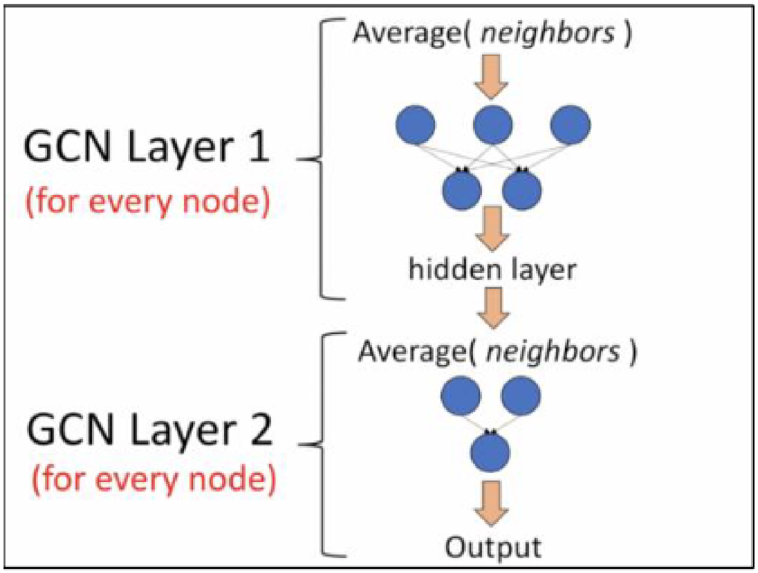


Figure 2.8: Example of a 2-layer GCN.

- **Linear Transformation:**

  Following the feature propagation step, a linear transformation is applied to the updated node features. This transformation aims to learn a more expressive and task-specific representation by mapping the aggregated features to a new feature space. The linear transformation is typically implemented as a matrix multiplication between the updated features and a learnable weight matrix. The weight matrix captures the relationships between the input features and the desired output [57].

- **Application of a Non-linear Activation Function:**

  To introduce non-linearity and capture more complex patterns, an activation function is applied element-wise to the transformed features. This non-linear mapping allows the GCN to model non-linear relationships between features and enables the network to learn more expressive representations. Common activation functions include the rectified linear unit (ReLU), sigmoid, or hyperbolic tangent (tanh) [57].

3.6.4 TYPES OF GCNS

Graph Convolutional Networks (GCNs) are classified into two primary categories: spectral-based GCNs and spatial-based GCNs [57].

- **Spectral-based GCNs:**

  Spectral-based GCNs were introduced by Bruna et al. [12], operating in the spectral domain by leveraging the graph Laplacian eigen basis. This methodology utilizes the graph Fourier transform to translate graph signals into the spectral domain, enabling the application of convolutions. By leveraging the eigenvalues and eigenvectors of the graph Laplacian, spectral-based GCNs effectively capture the global structure of the graph. The convolution operation in spectral-based GCNs can be defined as[57]:

  $$Y = \sum_{k=0}^{K} U \Lambda_k U^\top X W_k$$

  where:

  - Y is the output feature matrix,
  - X is the input feature matrix,
  - W_k represents the trainable weight matrix for the k-th graph convolutional layer,
  - U and Λ_k denote the eigenvectors and eigenvalues of the graph Laplacian, respectively,
  - K is the number of layers

- **Spatial-based GCNs:** also known as neighborhood aggregation or message-passing GCNs, operate in the spatial domain by aggregating information from the local neighborhood of each node. These models propagate information through message passing between neighboring nodes, effectively capturing the local structure and relationships within the graph. Spatial-based GCNs update node features by aggregating and transforming the features of neighboring nodes. The operation of spatial-based GCNs can be described by the formula:

  $$Y = \sigma(\mathbf{D}^{-1}\mathbf{AXW})$$

where:

- Y is the output feature matrix,
- X is the input feature matrix,
- W represents the trainable weight matrix,
- $\mathbf{A}$ denotes the adjacency matrix of the graph,
- $\mathbf{D}$ is the degree matrix, which is a diagonal matrix with the node degrees as its diagonal entries,
- σ represents the activation function.

These categories illustrate how GCNs adapt convolutional techniques for graphs, each focusing on different aspects of graph topology and connectivity to perform effective graph-based learning tasks[57].

4 Related Work

Recent advancements in session-based recommender systems (SBRS) have leveraged both graph neural networks (GNNs) and knowledge graphs (KGs) to enhance recommendation accuracy. In this section, we review the existing literature, focusing on GNN-based approaches and KG-based session-based recommender systems.

4.1 GNN Session-based Recommendation Systems

GNN-based session-based recommendation systems (SBRS) have made significant strides in recent years. Models like SR-GNN [63] and Star-GNN [41] are prominent examples in this domain. SR-GNN represents session sequences as graphs with items as nodes and transitions as edges, effectively capturing complex item transitions using gated GNNs. This approach allows for a nuanced understanding of item relationships within a session, significantly improving recommendation accuracy. However, SR-GNN does not incorporate external knowledge bases, limiting its ability to understand item relationships beyond session data. Additionally, it struggles with identifying fine-grained user-item interaction intents and maintaining long-range connectivity semantics within the session graph [63].

On the other hand, Star-GNN simplifies the session graph structure into a star graph, improving computational efficiency while maintaining essential interaction information. This model addresses some of the computational limitations faced by more complex graph structures. Despite these improvements, Star-GNN faces scalability challenges with very large datasets and may lose detailed interaction information, potentially reducing recommendation precision in complex scenarios [41].

4.2 Knowledge Graph-based Session-based Recommendation Systems

Knowledge graph-based session-based recommender systems (KG-SBRS) have enhanced recommendation accuracy by incorporating rich contextual information from knowledge graphs (KGs). Traditional models like RNNs and Markov Chains capture sequential dependencies but often fail to account for broader contextual relationships across sessions [62]. Integrating KGs with graph neural networks (GNNs) addresses these limitations, as seen in models like KGAT-SR[66] and SR-GNN, which leverage KGs to augment item features and utilize GNNs to capture complex item relationships within sessions [66]. However, these models often fail to adaptively propagate information according to session-specific contexts, leading to weakened relational data and suboptimal recommendations [62].

Recent models like Knowledge Graph-based Session Recommendation with Session-Adaptive Propagation build KGs with multi-typed edges to represent various user-item interactions and adaptively aggregate item neighbor information based on the specific intent of each session [62]. These approaches typically use user-item KGs to model user intents, aiming to understand the broader context of user interactions and enhance recommendation accuracy [61]. Despite these improvements, many approaches do not explicitly model user intents in a way that is adaptive to session-specific contexts.

5 Conclusion:

This chapter has outlined the fundamental principles of Knowledge Graphs (KGs) and Graph Convolutional Networks (GCNs), highlighting their significance in recommendation systems. Knowledge Graphs serve as structured frameworks that organize entities and their relationships, providing a rich context for understanding user preferences and item attributes. Graph Convolutional Networks leverage these structured frameworks to capture complex interactions within the data, enhancing the system's ability to make accurate and personalized recommendations. This chapter sets the groundwork for further exploration of the methodologies and practical applications of KGs and GCNs in the following chapters.

CHAPTER 3

Methodology For Knowledge Graph-based Intent for Session-based Recommendation System

1 INTRODUCTION

This chapter presents the methodology used to develop the Knowledge Graph-based Intent for Session-based Recommendation (KGIS) model. KGIS combines knowledge graphs (KGs) with session data to enable personalized recommendations in complex recommender systems. The chapter begins by defining basic concepts such as items, sessions, and knowledge graphs, laying the foundation for the subsequent methodology. It then outlines the systematic approach of KGIS, including how intents are extracted from KGs aligned with session items, how Graph Convolutional Networks (GCNs) are used to generate recommendations based on these intents, and how the model provides explanations for its recommendations. Each step is explained clearly, focusing on the theoretical framework. The chapter aims to provide a thorough understanding of how KGIS utilizes semantic relationships and user behavior patterns to improve the recommendation process.

2 CONCEPTS, NOTATIONS AND PROBLEM FORMULATION

2.1 ITEM

The set of all items in a recommender system is denoted as $V = \{v_1, v_2, \ldots, v_{|V|}\}$

2.2 SESSION

Session is a non-empty bounded list of interactions taken by a user. In this study, we use ordered, single-type-action, and anonymous sessions. Formally, a session of length l is denoted as sequence $S = \{v_1, v_2, \ldots, v_l\}$, which may contain duplicate items, and each item v_i belongs to the set of items.[66]

2.3 KNOWLEDGE GRAPH

A knowledge graph (KG) is defined as a collection of triples $G = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$. Each triple (h, r, t) indicates that a relation $r \in \mathcal{R}$ exists from the head entity $h \in \mathcal{E}$ to the tail entity $t \in \mathcal{E}$. For example, the triple (Martin Freeman, star, The Hobbit) describes that Martin Freeman is the star of the movie The Hobbit. Each entity in the entity set $\mathcal{E}$ is represented as a node in the graph, and each relation in the relation set $\mathcal{R}$ is represented as an edge in the graph. KGs are able to profile items and offer complementary information to the interaction data. We align the entity $e \in \mathcal{E}$ in the graph G with the item $v_i \in S$ in the session S to create the item entity set in the graph, denoted as $A = \{e \mid e \in \mathcal{E}$ can be aligned with $v_i \in S\}$ [61].

2.4 TASK DESCRIPTION

Given a session context of an ordered, single-type-action, and anonymous session S of length l and a knowledge graph G aligned with the items in the session, the next interaction (item) recommendation task is to identify the next possible interaction (item) v_{i+1} in S. The goal is to recommend a set of items V that the user is likely to be interested in, associated with reasoning.[66]

3 OUR PRPOSED MODEL

We now present the proposed Knowledge Graph-based Intent for Session-based (KGIS).The Figure 3.1 illustrates the overall framework of KGIS. It consists of three key steps:

Extracting intents: This step involves first mapping the items in the session to the KG to find their associated entities. p-hop neighbors of these associated entities are extracted from the KG to form a subgraph. Generates a score over all intents in the p-hop neighbors subgraph. The intent with the highest score is the best intent.

Generating recommendations: Using these intents, another graph is created where the intents are connected with items that have a relation with them in the KG. This graph is then used with a GCN model to generate recommendations.

Explainable recommendation: Using these intents, the system can offer clear and interpretable explanations for why a particular recommendation was made.

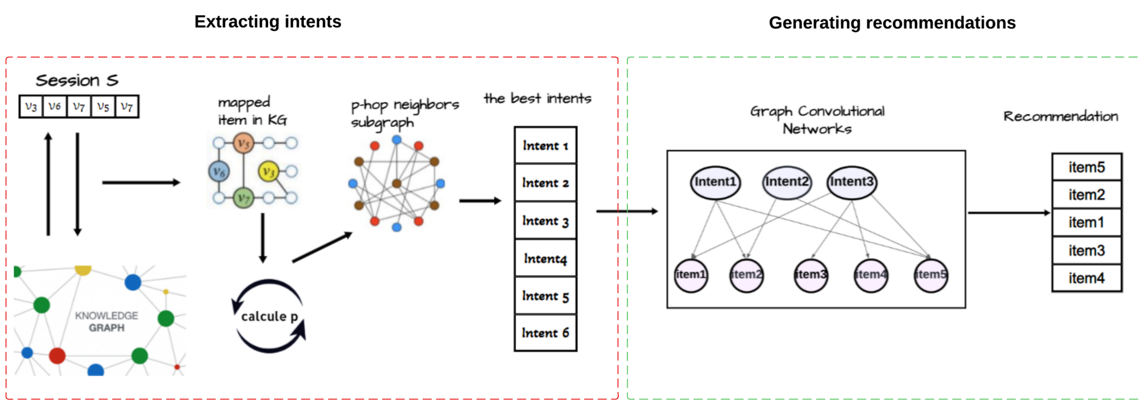


Figure 3.1: The overall framework of KGIS

3.1 EXTRACTING INTENTS

User behaviors are influenced by various intents, as defined in the paper "Learning Intents behind Interactions with Knowledge Graph for Recommendation" [61]. Here, intent refers to the underlying reasons guiding users' choices, revealing consistent patterns in their behaviors. For instance, in movie recommendations, intents could revolve around factors like actor combinations or directorial genres. Each intent signifies a distinct pattern of user behavior.

Our objective is to identify the intent i_s from the knowledge graph (KG) that best aligns with session S. It's crucial to recognize that S might include noisy items that are not derived from the true intent i_s but are correlated with positive items. This type of sampling is referred to as mixed sampling in inductive generalization [40], where some items in S are generated from the true intent (strong sampling) and others are not (weak sampling). However, since i_s is unknown, positive items and noisy items are indistinguishable, making it impossible to make assumptions about the sampling process of S.

This step consists of the following five key components:

1. **At first, we aim to determine the extent of subgraph neighbors, p, as a function of the variability of items in Session (S)**. To do so:

   (a) We first add for each item (V) from session (S), a vertex is joined to the node that corresponds to the entity to which it is mapped in the knowledge graph (KG). Notice that repetitive items are represented by different vertices.
   Consider $S = \{v_1, v_2, \ldots, v_n\}$ a set of items mapped to nodes $c_1, c_2, \ldots, c_k$ in the knowledge graph (KG).

   (b) Then for each pair of items x, y in S, the distance $d(x, y)$ is computed.

   (c) The obtained values are after that sorted in ascending order and the median (M) is detected. This latter is a good representative of the variability of the items. It gives us a better indication about the average distance between them since unlike the mean, it is less affected by outlier values.

   (d) The extent p is finally defined as $p = \lceil M \rceil$. By doing so, we are on one hand, defining the extent p of subgraph neighbors (SNs) as a function of the variability of S and on the other hand, neglecting outlier values that the distance could have, since noisy items in S may be semantically distant from positive items.

2. **Constructing The p-hop neighbors subgraph**:

   The p-hop neighbors subgraph for the session is the set of nodes (SNs), defined as:

$$SNs = \{c_1, c_2, \ldots, c_k\} \cup \left[\bigcup_{i=1}^{k} (D_p(c_i) \cup A_p(c_i)) \right] \tag{3.1}$$

   where $D_p(c_i)$ and $A_p(c_i)$ denote the set of nodes reachable from and to c_i within a path length of p in the knowledge graph (KG).

   Thus, the set $D_p(c_i) \cup A_p(c_i)$ corresponds to intents that generalize items from S that are mapped to c_i and which are at most p semantically distant from the latter.

   By identifying p-hop neighbors subgraph $(p - SNs)$ for S, we will considerably reduce the time of the search for the best intents.

3. **Extract the subgraph G_s:**

   For the next component, we need to extract the subgraph G_s induced by $(S) \cup (p - SNs)$ from p-hop neighbors subgraph where each item from S, is joined by an arc to the node of its corresponding entity.

4. **Score Intents Prediction:**

   Recall that our objective is to identify the intent i_s from p-hop neighbors subgraph (SNs) where no assumptions are made about the sampling process of the session. For each intent $i \in SNs$, we split the session into two subsets S_i^+ which denote respectively, items that satisfy i (positive items for i), and S_i^- denotes the items that do not (noisy items for i).

   The main idea of the approach is to generate score distribution over all intents in the p-hop neighbors subgraph, denoted $f(i)$, as a function of S_i^+ and S_i^-. We propose to formulate $f(i)$ as the difference between $f(S_i^+)$ and $f(S_i^-)$:

   $$f(i) = f(S_i^+) - f(S_i^-) \tag{3.2}$$

   where:

   $$f(S_i^+) = \sum_{v \in S_i^+} \frac{1}{d_{G_s}(v, i)} \tag{3.3}$$

   $$f(S_i^-) = \sum_{v \in S_i^-} \frac{1}{d_{G_s}(v, i)} \tag{3.4}$$

5. **The best intents:**

   The intent that fits the session the besti_s then deduced as:

   $$i_s = \arg\max_i f(i) \tag{3.5}$$

3.2 GENERATING RECOMMENDATIONS

After identifying and ranking the intents, we select the top K intents. Using these top intents, a graph is created where the intents are connected to items that have a relation with them in the KG. This graph is then used with a GCN model with one layer.

$$Z = GCN(X, A) \tag{3.6}$$

Let X represent the feature matrix of node embeddings and A denote the adjacency matrix capturing the graph structure.
The output Z consists of updated embeddings that captures the relationships and context within the graph. These refined embeddings are then used to identify relevant items by clustering them based on their similarity. The first cluster, which includes the items most aligned with user intents, is selected as the final set of recommendations.

3.3 EXPLAINABLE RECOMMENDATION

After identifying and selecting the top K intents, these are used to provide interpretable explanations of the most influential factors in making the recommendation. These explanations focus on the attributes of the items that led to the recommendation.

4 CONCLUSION

In this chapter, we outlined the framework and principles behind our Knowledge Graph-based Intent for Session-based Recommendation (KGIS) model. Our approach aims to enhance the accuracy and interpretability of recommendations by leveraging knowledge graphs and session data.

In the next chapter, we will detail the experiments conducted to evaluate our model, presenting the results and discussing their implications. We will also compare our findings with baseline methods to highlight the effectiveness of our approach.

CHAPTER 4

Experiment and Results

1 Introduction

In this chapter, we delve into the presentation and analysis of the final results obtained from our study. Supported by visualizations and detailed discussions, we explore the performance metrics, insights gleaned, and the broader implications of our findings within the context of personalized recommendation strategies.

2 Experimental Setup

2.1 Dataset Description

The MovieLens-1M[1] dataset is a stable benchmark dataset in the field of recommender systems and machine learning, provided by the GroupLens Research lab at the University of Minnesota. It contains one million movie ratings from 6,040 users on 3,952 movies[21]. Here is a detailed description:

Rating: 1,000,209 ratings (from 1 to 5) from 6,040 users on 3,952 movies.

Users: Each user has rated at least 20 movies. User data includes demographic information such as age, gender, occupation, and zip code.

[1] available at: https://grouplens.org/datasets/movielens/1m/

Movies: Each movie has a unique identifier, along with title and release year. Movies are categorized into one or more genres, from a predefined set of 18 genres.

2.2 Knowledge Graph

MovieLens-1M the corresponding KG has been extracted by Cao et al [13] gathers relevant facts from DBPedia, where the triplets are directly associated with the entities through mapped items, no matter which role (i.e. subject or object) the entity serves as. The preprocessed version used from us for our SIGIR22 paper [10]. This process yielded a knowledge graph characterized by:

- **Entities**: 13,822
- **Triples**: 323,499
- **Relation Types**: 18

2.3 Preprocessing

The dataset underwent extensive preprocessing steps inspired by prior work [64]:

- We filtered out the items in the datasets that have no corresponding entities in the KGs.
- users with fewer than 10 interactions were filtered out to focus on more active users.
- Interaction data were sorted chronologically by timestamp to preserve the temporal order of user-item interactions.
- Sessions were constructed by segmenting the sorted interactions based on a minimum session length of 3 interactions, a maximum length of 10 interactions, and a time threshold of 1 hour between consecutive interactions within a session.
- Sessions containing only 1 item and items occurring fewer than 5 times across the dataset were excluded to ensure robustness and relevance in session-based recommendation modeling.

- The preprocessed dataset was split into training (80%) and testing (20%) sets to facilitate model development and performance evaluation.

Table 4.1: Dataset Statistics

Dataset	**MovieLens-1M**
Users	6,040
Items	3,240
Ratings	940,963
Sessions	926,283
Avg. Session Length	4.05 interactions/session
Timestamp Range	April 25, 2000, to February 28, 2003

2.4 Experimental Design

2.4.1 Baseline Models

The study includes benchmarking against established baseline models:

1. **SKMeans (Spherical k-means based recommender):** Adjusts clustering weights dynamically to improve collaborative filtering[50].

2. **Item-Based Nearest Neighbor:** Identifies item relationships to enhance scalability and recommendation quality[51].

3. **Bayesian Personalized Ranking (BPR):** Tailored for personalized ranking using implicit feedback data to optimize item rankings[46].

4. **SR-GNN (Session-Based Recommendation with Graph Neural Networks):**Models session sequences as graphs using GNNs to capture intricate item transitions, improving session-based recommendations[63].

2.4.2 PROPOSED MODEL

The study proposes an advanced recommendation model that integrates item knowledge graphs with Graph Convolutional Networks (GCNs) to extract user intents and enhance recommendations. We select three intents to create the top 3 graphs and five intents to create the top 5 graphs, and train the model with them.
The process of creating the top-k intent graphs was computationally intensive, requiring 62 hours to generate 1,250 intent graphs (1,000 for training and validation, 250 for testing). This substantial time investment underscores the complexity of the graph construction process but is justified by the significant performance gains observed in the results.

2.5 EVALUATION METRICS

Model performance was evaluated using standard recommendation metrics:

- **HitRate@20**: Proportion of correctly recommended items within the top 20 recommendations[1].

$$\text{HitRate@20} = \frac{\text{Number of hits in top 20}}{\text{Total number of test cases}} \tag{4.1}$$

- **Recall@20**: Proportion of relevant items retrieved within the top 20 recommendations[1].

$$\text{Recall@20} = \frac{\text{Number of relevant items in top 20}}{\text{Total number of relevant items}} \tag{4.2}$$

3 RESULTS

3.1 TRAIN AND TEST LOSS

The train and validation loss plots for our model configurations (TOP3 and TOP5) are shown below:

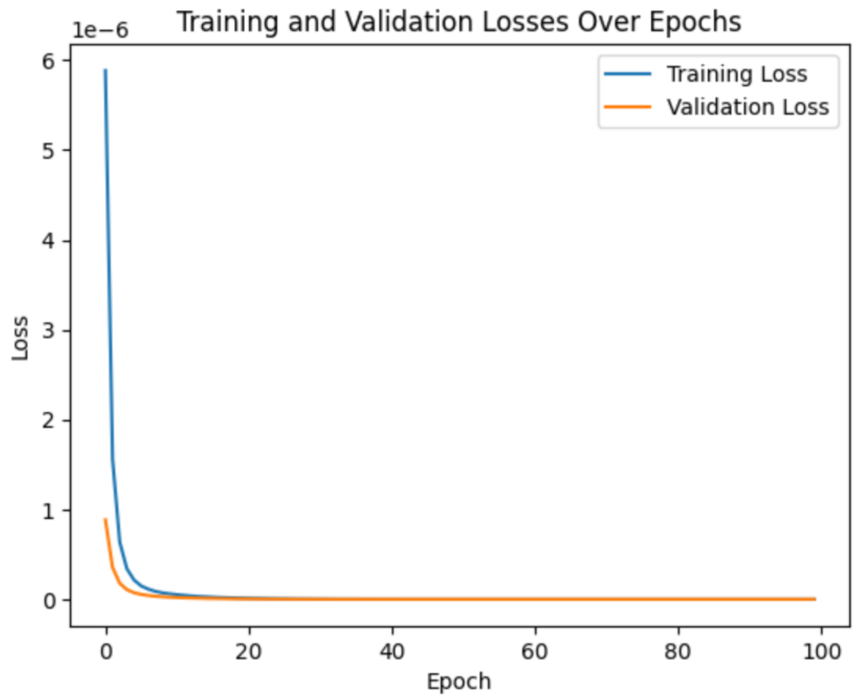


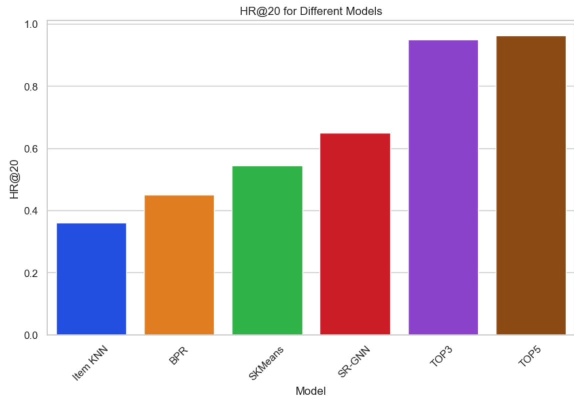
Figure 4.1: Train and Validation Loss for TOP3 and TOP5

3.2 HitRate@20 and Recall@20

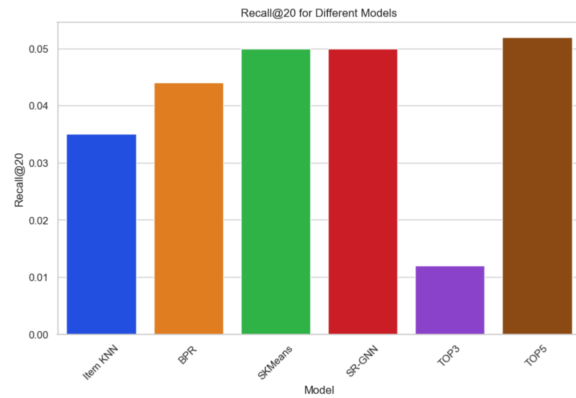
The performance of our model compared to the baseline models in terms of HitRate@20 and Recall@20 is summarized in the table and histogram plots below.

Table 4.2: Performance Metrics: HitRate@20 and Recall@20

Model	**HR@20**	**Recall@20**
Item KNN	0.36	0.035
BPR	0.45	0.044
SKMeans	0.545	0.05
SR-GNN	0.65	0.05
TOP3	0.95	0.012
TOP5	0.964	0.052



: HitRate@20 Histogram



: Recall@20 Histogram

Figure 4.2: Comparison of Performance Metrics: HitRate@20 and Recall@20 Histograms

3.3 DISCUSSION

The results clearly indicate that our proposed model, particularly the TOP5 , significantly outperforms traditional recommendation models in terms of both HitRate@20 and Recall@20. The integration of the knowledge graph and GCNs enables our model to capture complex user intents and item relationships more effectively, leading to more personalized and accurate recommendations.

The TOP3 model also performed well, especially in HitRate@20, although its Recall@20 was lower than the TOP5 model. This suggests that while the TOP3 model is good at placing relevant items within the top 20, the TOP5 model is better at ensuring a broader range of relevant items is included.

The training and validation loss plots indicate that our models converge well during training, with minimal overfitting observed. This further confirms the robustness and generalizability of our approach. Moreover, our model offers an added advantage of explainability. By leveraging the knowledge graph, the model can provide insights into why certain items are recommended. For example, consider the recommendation provided below:

4 Conclusion

In this chapter, we presented the experimental setup and methodology used to enhance recommendation systems through knowledge graph integration. We use a knowledge graph by combining entities from the MovieLens-1M dataset with DBpedia entities, enriching the dataset with semantic relationships. The preprocessing steps followed prior research guidelines, filtering out low-frequency users and items to ensure data quality.

Benchmarking against established models highlighted the effectiveness of our approach, which leveraged Graph Convolutional Networks (GCNs) on top of item knowledge graphs. Evaluation metrics like HitRate@20 and Recall@20 demonstrated superior performance compared to traditional methods.

Overall, our study contributes to advancing recommendation systems by integrating knowledge graphs and emphasizes the importance of explainability in enhancing user trust and satisfaction.

```
I RECOMMENDED THIS MOVIE 'Mission:_Impossible_(film)' BECAUSE YOU APPEAR TO LIKE:

Category:American_films
List_of_film_spoofs_in_Mad
Category:English-language_films
List_of_action_films_of_the_1990s
```

Explainable Recommendation Example

Mission impossible

MISSION: IMPOSSIBLE

Logo original du film

Titre original	*Mission: Impossible*
Réalisation	Brian De Palma
Scénario	David Koepp Robert Towne
Musique	Danny Elfman
Acteurs principaux	Tom Cruise Emmanuelle Béart Jon Voight Jean Reno
Sociétés de production	Cruise/Wagner Productions Paramount Pictures
Pays de production	États-Unis
Genre	Espionnage
Durée	110 minutes
Sortie	1996

Film Side Information (Wikipedia)

Figure 4.3: Example of Explainable Recommendation and Side Information

General Conclusion

This thesis has endeavored to advance the state-of-the-art in recommendation systems, focusing on enhancing user understanding, recommendation accuracy, and explainability while acknowledging computational challenges. Through the exploration and development of our model, several key findings and contributions have emerged.

Firstly, our model has demonstrated a robust capability to understand user behavior and preferences. By leveraging knowledge graphs and employing advanced machine learning techniques, particularly Graph Convolutional Networks (GCNs), we have effectively captured intricate patterns in user-item interactions. This understanding enables our model to recommend items that align closely with user preferences, thereby enhancing user satisfaction and engagement.

Moreover, a pivotal aspect of our contribution lies in the model's ability to provide explainable recommendations. Despite the inherent challenge of limited user information, our approach integrates interpretability mechanisms that elucidate the reasoning behind each recommendation. This transparency not only enhances user trust but also empowers users to comprehend and potentially refine their preferences.

Furthermore, our model excels in diversifying recommendations by introducing users to novel items that resonate with their historical preferences. This capability is crucial in mitigating the issue of recommendation stagnation and ensuring that users encounter new and relevant content tailored to their evolving tastes.

However, it is essential to acknowledge the limitations encountered during this research. The computational costs associated with processing and analyzing large-scale knowledge graphs posed significant challenges. Specifically, constructing and utilizing extensive graphs for training and inference consumed substantial computational resources and time. These constraints hindered further exploration and optimization across additional performance metrics. Additionally, our model faces a limitation in its inability to rank items within clusters, which affects the precision of recommendations when dealing with grouped items.

Looking ahead, future research directions should aim to address these limitations while continuing to advance the efficacy and scalability of recommendation systems. Improvements in computational efficiency, possibly through innovative graph processing techniques or hardware advancements, could unlock greater potential in model performance and scalability. Additionally, exploring hybrid approaches that integrate diverse data sources or leveraging reinforcement learning paradigms could further enrich the recommendation capabilities.

In conclusion, this thesis contributes to the field by showcasing a model that not only comprehends user behavior and provides accurate recommendations but also ensures transparency through explainable recommendations. Despite computational challenges, the findings underscore the potential of integrating knowledge graphs and advanced machine learning for personalized recommendation systems.

Bibliography

[1] 10 metrics to evaluate recommender and ranking systems — evidentlyai.com. https://www.evidentlyai.com/ranking-metrics/evaluating-recommender-systems. [Accessed 09-06-2024].

[2] A Comprehensive Introduction to Graph Neural Networks (GNNs) — datacamp.com. https://www.datacamp.com/tutorial/comprehensive-introduction-graph-neural-networks-gnns-tutorial. [Accessed 12-05-2024].

[3] Big data recommendation systems — linkedin.com. https://www.linkedin.com/pulse/big-data-recommendation-systems-jahnavi-redrouthu/. [Accessed 1-05-2024].

[4] Home - DBpedia Association — dbpedia.org. https://www.dbpedia.org/. [Accessed 16-04-2024].

[5] What is a Recommendation System? — nvidia.com. https://www.nvidia.com/en-us/glossary/recommendation-system/. [Accessed 1-06-2024].

[6] Wikidata — wikidata.org. https://www.wikidata.org/wiki/Wikidata:Main_Page. [Accessed 16-04-2024].

[7] Graph neural network and its applications, 2022.

[8] Bilal Abu-Salih. Domain-specific knowledge graphs: A survey. *Journal of Network and Computer Applications*, 185:103076, 2021.

[9] Charu C Aggarwal. *Recommender Systems: The Textbook*. Springer, 2016.

[10] Giacomo Balloccu, Ludovico Boratto, Gianni Fenu, and Mirko Marras. Post processing recommender systems with knowledge graphs for recency, popularity, and diversity of explanations. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 646–656, 2022.

[11] Kurt Bollacker, Robert Cook, and Patrick Tufts. Freebase: A shared database of structured general human knowledge. In *AAAI*, volume 7, pages 1962–1963, 2007.

[12] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[13] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*, pages 151–161, 2019.

[14] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. *arXiv preprint arXiv:1611.03954*, 2016.

[15] Penghe Chen, Yu Lu, Vincent W Zheng, Xiyang Chen, and Boda Yang. Knowedu: A system to construct knowledge graph for education. *Ieee Access*, 6:31553–31563, 2018.

[16] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PloS one*, 10(6):e0128193, 2015.

[17] Jeffrey Dalton, Laura Dietz, and James Allan. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 365–374, 2014.

[18] Sanket Doshi. Brief on recommender systems — towardsdatascience.com. https://towardsdatascience.com/brief-on-recommender-systems-b86a1068a4dd. [Accessed 12-06-2024].

[19] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.

[20] Christian Grévisse, Rubén Manrique, Olga Mariño, and Steffen Rothkugel. Knowledge graph-based teacher support for learning material authoring. In *Advances in Computing: 13th Colombian Conference, CCC 2018, Cartagena, Colombia, September 26–28, 2018, Proceedings 13*, pages 177–191. Springer, 2018.

[21] GroupLens. MovieLens 1M Dataset — grouplens.org. https://grouplens.org/datasets/movielens/1m/. [Accessed 23-05-2024].

[22] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3549–3568, 2020.

[23] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3549–3568, 2022.

[24] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2016.

[25] Zhiyong Huang and W Bruce Croft. A unified relevance model for opinion retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 947–956, 2009.

[26] Zhiyuan Huang, Zhi Zhang, and Chuan Li. Knowledge graph embedding based personalized recommendation. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 21–28. IEEE, 2018.

[27] IBM. What is a knowledge graph? https://www.ibm.com/topics/knowledge-graph, April 2021. Accessed: 2024-05-29.

[28] Folasade Olubusola Isinkaye, Yetunde O Folajimi, and Bolande Adefowoke Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, 16(3):261–273, 2015.

[29] Yan Jia, Yulu Qi, Huaijun Shang, Rong Jiang, and Aiping Li. A practical approach to constructing a knowledge graph for cybersecurity. *Engineering*, 4(1):53–60, 2018.

[30] Shah Khusro, Zafar Ali, and Irfan Ullah. Recommender systems: issues, challenges, and research opportunities. In *Information science and applications (ICISA) 2016*, pages 1179–1189. Springer, 2016.

[31] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2017.

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

[33] Vijaysinh Lendave. Cold-start problem in recommender systems and its mitigation techniques, 2021.

[34] Jue Liu, Zhuocheng Lu, and Wei Du. Combining enterprise knowledge graph and news sentiment analysis for stock price prediction. 2019.

[35] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. *Content-based Recommender Systems: State of the Art and Trends*, pages 73–105. Springer US, Boston, MA, 2011.

[36] Yukun Ma, Haiyun Peng, and Erik Cambria. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive lstm. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[37] Sameh K Mohamed, Aayah Nounu, and Vít Nováček. Biological applications of knowledge graph embedding models. *Briefings in bioinformatics*, 22(2):1679–1693, 2021.

[38] Diego Moussallem, Matthias Wauer, and Axel-Cyrille Ngonga Ngomo. Machine translation using semantic web technologies: A survey. *Journal of Web Semantics*, 51:1–19, 2018.

[39] Muvi. Hybrid recommendation system – a beginner's guide. https://www.muvi.com/resources/ebooks/hybrid-recommendation-system/, May 2022. Accessed: 2024-05-23.

[40] Daniel J Navarro, Matthew J Dry, and Michael D Lee. Sampling assumptions in inductive generalization. *Cognitive Science*, 36(2):187–223, 2012.

[41] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten De Rijke. Star graph neural networks for session-based recommendation. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1195–1204, 2020.

[42] Subhashree Panda. *Recurrent unit together with reinforcement learning for graph networks*. Master thesis, 2023.

[43] Jianwei Qian, Xiang-Yang Li, Chunhong Zhang, Linlin Chen, Taeho Jung, and Junze Han. Social network de-anonymization and privacy inference with knowledge graph model. *IEEE Transactions on Dependable and Secure Computing*, 16(4):679–692, 2017.

[44] Ruihong Qiu, Zi Huang, Tong Chen, and Hongzhi Yin. Exploiting positional information for session-based recommendation. *ACM Transactions on Information Systems (TOIS)*, 40(2):1–24, 2021.

[45] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. Sequence-aware recommender systems. *ACM computing surveys (CSUR)*, 51(4):1–36, 2018.

[46] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[47] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Introduction to Recommender Systems Handbook*, pages 1–35. Springer US, Boston, MA, 2011.

[48] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. *Recommender Systems Handbook*. Springer, 2010.

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

[50] Aghiles Salah, Nicoleta Rogovschi, and Mohamed Nadif. A dynamic collaborative filtering system via a weighted clustering approach. *Neurocomputing*, 175:206–215, 2016.

[51] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

[52] Simon Schramm, Christoph Wehner, and Ute Schmid. Comprehensible artificial intelligence on knowledge graphs: A survey. *Journal of Web Semantics*, 79:100806, 2023.

[53] Lalita Sharma and Anju Gera. A survey of recommendation system: Research challenges. *International Journal of Engineering Trends and Technology (IJETT)*, 4(5):1989–1992, 2013.

[54] Longxiang Shi, Shijian Li, Xiaoran Yang, Jiaheng Qi, Gang Pan, and Binbin Zhou. Semantic health knowledge graph: semantic integration of heterogeneous medical knowledge and services. *BioMed research international*, 2017(1):2858423, 2017.

[55] Thiago Silveira, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. How good your recommender system is? a survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, 10:813–831, 2019.

[56] Mingsheng Sun, De Li, Jun Wang, Jianyong Li, and Shaojie Wang. Knowledge graph embedding and learning through link prediction. *arXiv preprint arXiv:1804.07847*, 2018.

[57] YAHIAOUI Somia TOUHAMI Hadil. DSpace de l'universit&xE9; Ibn khaldoun de Tiaret: Session-based Recommendation Systems with Graph ATtention Networks — dspace.univ-tiaret.dz. http://dspace.univ-tiaret.dz:80/handle/123456789/13441, 2023. [Accessed 07-05-2024].

[58] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 1835–1844, 2018.

[59] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z Sheng, Mehmet A Orgun, and Defu Lian. A survey on session-based recommender systems. *ACM Computing Surveys (CSUR)*, 54(7):1–38, 2021.

[60] Xiang Wang, Xiangnan He, Miao Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–174, 2019.

[61] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the web conference 2021*, pages 878–887, 2021.

[62] Yu Wang, Amin Javari, Janani Balaji, Walid Shalaby, Tyler Derr, and Xiquan Cui. Knowledge graph-based session recommendation with session-adaptive propagation. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 264–273, 2024.

[63] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353, 2019.

[64] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 285–294, 2019.

[65] Dehai Zhang, Menglong Cui, Yun Yang, Po Yang, Cheng Xie, Di Liu, Beibei Yu, and Zhibo Chen. Knowledge graph-based image classification refinement. *IEEE Access*, 7:57678–57690, 2019.

[66] Qianqian Zhang, Zhuoming Xu, Hanlin Liu, and Yan Tang. Kgat-sr: knowledge-enhanced graph attention network for session-based recommendation. In *2021 IEEE*

33rd International Conference on Tools with Artificial Intelligence (ICTAI), pages 1026–1033. IEEE, 2021.

[67] Shuai Zhang, Lixin Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):1–38, 2019.

[68] Yueqin Zhu, Wenwen Zhou, Yang Xu, Ji Liu, and Yongjie Tan. Intelligent learning for knowledge graph towards geological data. *Scientific Programming*, 2017.

[69] Xiaohan Zou. A survey on application of knowledge graph. In *Journal of Physics: Conference Series*, volume 1487, page 012016. IOP Publishing, 2020.