SEMANTIC EXPLAINABLE RECOMMENDER WITH KNOWLEDGE GRAPH EMBEDDING AND DEEP REINFORCEMENT LEARNING

NEERAJ TIWARY

UNIVERSITI KEBANGSAAN MALAYSIA

SEMANTIC EXPLAINABLE RECOMMENDER WITH KNOWLEDGE GRAPH EMBEDDING AND DEEP REINFORCEMENT LEARNING

NEERAJ TIWARY

THESIS SUBMITTED IN FULFILMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

FACULTY OF INFORMATION SCIENCE AND TECHNOLOGY
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2024

PENCADANG SEMANTIK BOLEH DIJELASKAN BERASASKAN GRAF PENGETAHUAN TERBENAM DAN PEMBELAJARAN PENGUKUHAN MENDALAM

NEERAJ TIWARY

TESIS YANG DIKEMUKAKAN UNTUK MEMPEROLEH
IJAZAH DOKTOR FALSAFAH

FAKULTI TEKNOLOGI DAN SAINS MAKLUMAT
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2024

DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

04 July 2024

NEERAJ TIWARY
P105920

ACKNOWLEDGEMENT

With the grace of almighty God, I stand at this moment, humbled, and overwhelmed with gratitude, as I reflect on the incredible journey of my Ph.D. study. With utmost sincerity, I express my heartfelt thanks to God, who gave me immense power, and all the individuals who have supported me throughout this challenging but rewarding path.

I owe gratitude to my esteemed advisor, Prof. Dr. Shahrul Azman Mohd Noah. His guidance, mentorship, and open-minded approach have been instrumental in shaping my research and helping me navigate the complexities of academia. His profound insights, thoughtful feedback, and unwavering support have been the driving force behind the successful completion of my Ph.D. and this dissertation. Without his mentorship, this achievement would have remained an unattainable dream. I am also particularly grateful to Dr. Wan Fariza Fauzi, my co-supervisor, for her assistance and insightful feedback throughout my research work.

I am also deeply grateful to Prof. Steffen Stabb, whose constructive feedback on writing impactful articles has been invaluable. His guidance encouraged me to think deeply about my research topic and hone my skills in composing influential research papers. His thoughtful advice led me to contribute meaningfully to my field of study and pursue publication in esteemed journals.

To Dr. Tan Siok Yee, I extend my thanks for her support and valuable feedback on my submitted articles. Her constructive criticism and suggestions helped refine my work and broaden my perspective.

I would be remiss not to express my appreciation to all the staff at Universiti Kebangsaan Malaysia (UKM). From administrative support to research facilities, your assistance and dedication have eased many aspects of my academic journey. I am thankful to the university for providing the research grant that enabled me to submit articles to the premier journals. This support exemplifies the university's commitment to nurturing academic excellence and fostering meaningful research.

I thank my beloved family for their unconditional devotion and support. Their unwavering love, encouragement, and understanding have been the pillar of strength that sustained me through the highs and lows of this academic pursuit. Their belief in me, even during moments of self-doubt, pushed me to strive harder and inspired me to persevere.

Finally, to all those who have played a part in my academic journey, I am deeply grateful for your unwavering support and encouragement. Each of you has contributed to my growth as a scholar and an individual. I am forever thankful to everyone.

ABSTRAK

Sistem pencadang berasaskan pembelajaran mesin menunjukkan ketepatan signifikan dalam memberikan cadangan kepada individu. Walau bagaimanapun, kekurangan ketelusan dan keboleh tafsiran yang ditawarkan oleh model pembelajaran mesin ini menimbulkan cabaran dalam memahami penaakulan yang diberikan disebalik setiap cadangan. Graf pengetahuan atau knowledge graph (KG) memberi ruang untuk meningkatkan kemampuan keboleh-jelasan sistem pencadang dengan menjelaskan hubungan antara pilihan pengguna dan item yang dicadangkan. Walau bagaimanapun, aspek keboleh-jelasan masih merupakan bidang yang sangat baharu walaupun banyak penyelidikan sedang dijalankan. Penilaian kuantitatif kualiti keboleh-jelasan juga merupakan antara jurang penyelidikan. Kajian penyelidikan ini mencadangkan XAIRec, iaitu merupakan rangka kerja komprehensif untuk meningkatkan ketelusan dan keboleh-htafsiran dalam model sistem pencadang. XAIRec menggabungkan kelebihan bidang keboleh-jelasan kecerdasan buatan atau Explainability AI (XAI) , graf pengetahuan dan pembelajaran pengukuhan. Metod Guided Representation of User Preferences via the Path Embedding Propagation (RUPPEP) yang dicadangkan membentuk asas binaan, disamping menggunakan maklumat interaksi pengguna untuk mempelajari pemilihan. Dengan menggabungkan strategi ganjaran lunak, pemangkasan tindakan bersyarat pengguna dan pemarkahan berbilang lompatan, XAIRec menggunakan algoritma Max Explainability Score (MES) yang dipandu oleh rangkaian polisi untuk menentukan laluan penaakulan optimum dan set item calon. XAIRec seterusnya menggunakan algoritma keutamaan produk berasaskan Products Prioritisation Score (PPS) untuk meningkatkan kualiti pencadangan. Sistem Pencadang dengan utiliti keboleh-jelasan masih tidak mempunyai skor-metrik boleh ukur. Oleh itu kajian ini mencadagkan metrik boleh diukur untuk mengukur kualiti penjelasan yang dijana oleh sistem pencadang dipacu graf pengetahuan. Kajian ini memanfaatkan algoritma MES, iaitu metrik yang menilai kualiti penjelasan oleh sistem pencadang berasaskan graf pengetahuan. MES menggabungkan empat parameter penilaian utama: R (bilangan peraturan/cirian/lompatan), P (kebarangkalian laluan untuk item yang dicadang), $\mathbb{H}$ (entropi/nilai maklumat) dan Rw (ganjaran daripada laluan yang dipilih untuk produk yang dicadangkan). Algoritma MES menggunakan parameter ini dan menawarkan ukuran kualiti penjelasan yang sistematik dan bersifat boleh diukur. Kajian ini juga menilai satu lagi parameter penting, S (afiniti pengguna terhadap produk), dan menghasilkan algoritma PPS menggunakan parameter yang dinyatakan sebelum ini untuk menjana skor pengutamaan produk untuk pengguna. Penilaian menyeluruh merentasi pelbagai domain set data e-komers Amazon menunjukkan kekuatan XAIRec dalam menghasilkan laluan penaakulan yang boleh ditafsirkan untuk item yang dicadangkan. Dengan menggabungkan teknik terkini dari bidang model cadangan berasaskan XAI, kajian ini menyumbang kepada evolusi sistem pencadang bersifat boleh percaya.

ABSTRACT

Recommender systems (RSs) based on machine learning have demonstrated significant accuracy in providing personalized recommendations. However, the lack of transparency and interpretability offered by these models has posed challenges in understanding the reasoning behind recommendations. Knowledge graphs (KGs) provide an opportunity to enhance the explainability of RSs by revealing the underlying connections between user preferences and recommended items. Despite ongoing research, explainability remains a novice field, and the quantitative evaluation of the quality of these explanations is a significant research gap. This research study presents XAIRec, a comprehensive framework to enhance transparency and interpretability in recommendation models. XAIRec combines the strengths of Explainable AI (XAI), KGs, and reinforcement learning (RL). The Guided Representation of User Preferences via the Path Embedding Propagation (RUPPEP) method forms the foundation by developing an Actor-Critic RL model that utilizes historical interactions to learn user preferences. The XAIRec framework employs the RUPPEP algorithm to train a policy network by incorporating soft reward strategies, user-conditional action pruning, and multi-hop scoring. Subsequently, it applies the Max Explainability Score (MES) to determine optimal reasoning paths and candidate item sets. It further applies a Product Prioritisation Score (PPS)-based product prioritization algorithm to enhance recommendation quality. This study also addresses the need for quantifiable metrics to measure the quality of explanations generated by KG-driven RSs, which are currently lacking in Explainable RSs. The MES algorithm, an advanced metric that evaluates the quality of explanations provided by KG-based RSs, incorporates four key evaluation parameters: R (number of rules/features/hops), P (probability of traversal path for the recommended item), H (entropy/information value), and Rw (reward from chosen traversal paths for recommended products). The proposed MES algorithm leverages these parameters and offers a systematic and quantifiable measure of explanation quality. This study also evaluates another crucial parameter, S (user affinity for the product), and devises the PPS algorithm using the previously calculated parameters to capture the products prioritisation scores for the user. Extensive evaluations across various Amazon e-commerce domains demonstrate the prowess of XAIRec in generating interpretable reasoning paths for recommended items. By incorporating cutting-edge techniques from the field of XAI-driven recommendation models, this study contributes to the evolution of trustworthy RSs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF SYMBOLS AND TERMINOLOGY

Symbols	Descriptions
u_i	User i
v_j	Item j
e_k	Entity k in the knowledge graph
r_k	Relation between two entities (e_i, e_j) in the knowledge graph
$\mathcal{U} = \{u_1, \cdots, u_m\}$	User set
$\mathcal{V} = \{v_1, \cdots, v_n\}$	Item set
$\boldsymbol{u_{Embed}}$	Embedding vector of User $\mathcal{U}$
$\boldsymbol{v_{Embed}}$	Embedding vector of Item $\mathcal{V}$
$\mathcal{G}\text{know} = (V, E)$	Knowledge graph with vertex V and edges E
$\boldsymbol{\mathcal{P}}$	A meta-path in the knowledge graph connecting two entities
$\boldsymbol{\mathcal{P}_{MG}}$ $= \{\mathcal{P}_{1A_{0k}}, \mathcal{P}_{2A_{0k}}, \cdots, \mathcal{P}_{nA_{0k}}\}$	A meta-graph with many meta-paths in the KG connecting two entities
$\boldsymbol{\mathcal{P}_{MG}^{uv}}$	A meta-graph with meta-paths in the KG connecting user u_i to item v_j.
$I \in \mathbb{R}^{m \times n}$	An interaction matrix of users' interactions with the items
$e_H \in \mathcal{N}_{e_0}^H$	e_H is the H-hop neighbour of e_0
$p(E)$	Probability of an event E
$\mathbb{H}(E)$	Entropy or Information value of an event E
$\boldsymbol{\mathcal{H}}$	Candidate set of explainability of the recommended product for a user
$\mathbb{R}$	The number of rules/features/hops outputted by the explanation
$\mathbb{P}$	The probability of the traversal path for the recommended item
$\mathbb{H}$	The entropy/information value due for the recommended item
$\mathbb{S}$	The affinity score of the traversal path for the recommended item
$\mathbb{R}\mathbb{w}$	The reward of the traversal path for the recommended item.
$\odot$	Dot Product or Cosine similarity
$P@k$	Precision @ k (upto a ranking position k)
$R@k$	Recall @ k (upto a ranking position k)
$F@k$	F-Score @ k (upto a ranking position k)
MAP	Mean Average Precision
$nDCG@k$	Normalised Discounted Cumulative Gain @ k (upto a ranking position k)
MRR	Mean Reciprocal Rank
HR	Hit Rate
MEP	Mean Explainability Precision
MER	Mean Explainability Recall

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
CF	Collaborative Filtering
DARPA	Defense Advanced Research Projects Agency
DL	Deep Learning
GDPR	General Data Protection Regulation
HIN	Heterogeneous Information Network
HR	Hit Rate
IBCF	Item-based Collaborative Filtering
KG	Knowledge Graph
KGE	Knowledge Graph Embedding
LFM	Latent Factor Model
MF	Matrix Factorization
NDCG	Normalized Discounted Cumulative Gain
RDF	Resource Description Framework
RL	Reinforcement Learning
RS	Recommender System
UBCF	User-based Collaborative Filtering
UKM	Universiti Kebangsaan Malaysia
XAI	Explainable Artificial Intelligence
XAIRec	Explainable AI-Driven Recommender System
MDP	Markov Decision Process

CHAPTER I

INTRODUCTION

1.1 INTRODUCTION

With the exponential growth of online content, it has become increasingly difficult for users to navigate all the options and make informed decisions. This phenomenon is known as "information overload" (Tessier, 2020). The concept has a root in the idea that individuals exposed to more information than they can effectively process may hinder their insightful knowledge-processing capability and ultimately hamper their decisions making ability (Phillips-Wren & Adya, 2020).

Recommender systems (RSs) play a crucial role and can effectively address the problem of information overload. These systems use various algorithms and techniques to filter and present relevant content to users based on their preferences, behaviors, and historical data. The goal is to assist users in discovering content that aligns with their interests and needs (Wu et al., 2022). RSs are widely adopted in various industries, including e-commerce (Alamdari et al., 2020), streaming services (O'Dair & Fry, 2020), and social media (Pan et al., 2020), among many others. They help users find relevant products, movies, music, articles, and other items of interest to them.

While RSs have undoubtedly improved the efficiency of content discovery, they often operate as black boxes, where their inner workings and the reasoning behind their recommendations are not transparent or easily understandable to end-users. This lack of transparency in RSs poses significant challenges, including reduced user trust, potential bias, limited serendipity, and user engagement. This opacity hinders users' ability to understand the generated recommendations, limits

personalization, and raises concerns about accountability and fairness (Adadi & Berrada, 2018).

Addressing these issues requires the development of explainable algorithms, interpretable models, user-friendly interfaces, bias mitigation techniques, feedback loops, and user education to have a striking balance between algorithm complexity and user comprehension, fostering user confidence and more positive interaction with the systems (Evans et al., 2022).

The explainable recommendation (XR) is an emerging research area. It can potentially alleviate issues associated with opaque RSs by offering transparent and understandable rationales for their recommendations, fostering user trust and confidence. By providing insights into the decision-making process, explainable recommender systems (XRS) enable users to comprehend the basis for suggestions, promote exploration beyond filter bubbles, encourage user engagement, mitigate biases, and strike a balance between personalization and serendipity, ultimately enhancing the overall user experience while addressing challenges posed by traditional black-box models (Ehsan et al., 2021). The example below fully illustrates the aforementioned notion.

Major organizations like YouTube, Amazon, and Netflix possess extensive user-specific behavioural and transactional data, which they aim to leverage, as depicted in **Figure 1.1,** to guide current and potential users, fostering product promotion, cross-selling, and up-selling of content. Since these organizations have very few physical interactions with their end-users, they are dependent on technology to fill this gap. These organizations require a robust recommendation engine that comprehends the interests and needs of the users and recommends content suggestions accordingly. Recommendations can vary between generic and highly personalized, with the possibility of users seeking explanations for recommendations, thereby bolstering trust in the system. Progress in KGs, RL, and language models contribute to the pursuit of eXplainable Artificial Intelligence (XAI) in this context.

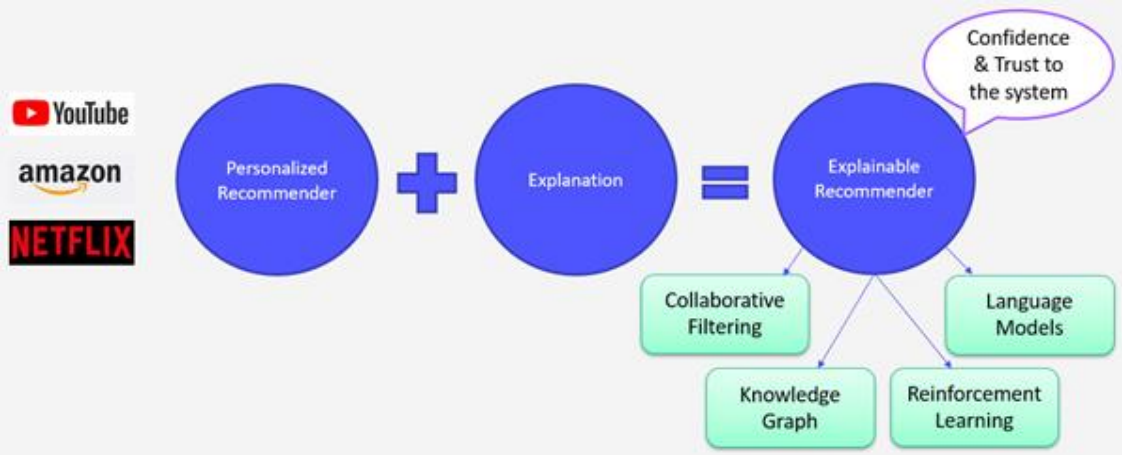

Figure 1.1 Explainable Recommender and its Implementation Approaches

This research aims to enhance the area of XRS to address issues about personalization, the transparency of reasoning, and the quantitative evaluation of explanatory efficacy. By building upon existing research, this research seeks to advance the comprehension of methods to enhance recommendation results, offer transparent rationales for these suggestions, and introduce robust metrics for appraising the explanatory value within RSs. Through these efforts, the study aspires to contribute to a more comprehensive understanding of improving recommendation processes, delivering more intelligible justifications for recommendations, and establishing dependable criteria for assessing the explanatory quality within the RS framework.

1.2 RESEARCH BACKGROUND

The exponential surge in the volume, diversity, and speed of data, driven by the rise of big data, has resulted in an overwhelming influx of information that users need to navigate. This deluge of information can pose a significant challenge for users trying to sift through and find pertinent and tailored content. To address this issue, RSs have emerged as a solution, offering personalized suggestions based on user preferences and behaviour. These systems play a crucial role across various domains, including music (Hu et al., 2018), movies (Zhang F. et al., 2016), online shopping (Zhao et al., 2017), and more. They employ various techniques, such as collaborative filtering

(CF), content-based filtering, and hybrid approaches, to generate these recommendations (Sharma et al., 2016).

The impact of RSs has been significant across various industries. These systems enhance user experience and engagement by offering personalised and relevant suggestions. This, in turn, boosts user satisfaction and loyalty, leading to increased revenue for organizations. For example, in the entertainment industry, platforms like Netflix (Amatriain & Basilico, 2015) and Spotify (Björklund et al., 2022) heavily rely on RSs to keep users engaged and coming back for more content. E-commerce platforms like Amazon and online advertising platforms use RSs to suggest products or ads tailored to individual users, thereby increasing the likelihood of conversions and sales (Rawat et al., 2021).

Thus, in today's era of information abundance, RSs have emerged as indispensable tools for tackling information overload by tailoring content and suggestions to individual user preferences. Their widespread integration across various sectors, such as e-commerce, entertainment, and content platforms, has elevated user satisfaction through personalized experiences and driven substantial business growth by fostering higher customer engagement, conversions, and revenue generation (Fayyaz et al., 2020).

RSs developed through techniques such as CF (Sahu & Saritha, 2021), matrix factorization (Liu & Zhao, 2023), and deep learning (DL) (Rostami et al., 2022) have achieved impressive accuracy in building these systems. However, a significant drawback of these approaches is their limited transparency and interpretability. They cannot explain why certain items got recommended out of other candidates (Zhang & Chen, 2020). As RSs play a growing role in influencing user decisions, it is crucial to address these concerns to establish user trust, explain recommendations, and mitigate potential bias. The issue has attracted various proposals for XRS models.

The enforcement of Artificial Intelligence (AI) regulations such as the European Union General Data Protection Regulation (GDPR) (Voigt & Von dem Bussche, 2017), the European Union AI Act 2023 (Laux et al., 2023), and the

California Consumer Privacy Act of 2018 (de la Torre, 2018), underscores the "right to explanation" in algorithmic decision-making, highlighting the growing importance of AI system explainability. This emphasis arises from concerns about transparency, accountability, bias mitigation, and ethical considerations. Overall, the explainability of AI systems has emerged as an imperative topic.

XR refers to personalised recommendation algorithms that deal with the issue of WHY, i.e., providing the reasoning behind the recommendations. It provides users with not just the suggested recommendations but also explainability, i.e., the explanations for those recommendations. It incorporates XAI into RSs, which has been found to increase user satisfaction. By incorporating XAI principles, these systems enable users to understand the rationale behind the suggestions, thus increasing user satisfaction and trust. XAI techniques, such as generating human-interpretable explanations, model visualization, and highlighting influential factors, help shed light on the decision-making process of AI models, making them more transparent and comprehensible to both users and experts in various domains. This transparency not only enhances user confidence in the recommendations but also assists regulators in ensuring the ethical and fair usage of AI systems (Ali et al., 2023).

Providing explainability for recommendations generated by RSs is crucial for building trust and understanding among users. There are generally two main approaches for achieving explainability in RSs.

The first approach adopts the model-agnostic or post-hoc methodologies. They focus on providing explanations for recommendations without altering the underlying recommendation algorithm. They adopt techniques such as feature importance, surrogate models, local reasoning, and SHAP Values. (Ai et al., 2018) proposed ECFKG to enhance the CF method over knowledge graph embedding (KGE) for personalized recommendation, followed by a soft matching algorithm to find explanation paths between users and items. However, one issue of this strategy is that the explanations are not produced according to the reasoning process but instead generated by an empirical similarity matching between the user and item embeddings.

Hence, their explanation component merely tries to find a post-hoc explanation for the already chosen recommendations.

The second approach adopts model-specific or, model-centric or model-intrinsic explainability. This approach involves designing recommendation algorithms that inherently provide explanations along with their recommendations. They adopt techniques such as explicit rules derived through rule-based systems that are easily understood by users, leveraging the structure of the recommendation network available through graph-based approaches. The graph-based approaches are also termed KG-based approaches.

Researchers have turned their attention to KG to tackle the challenges related to transparency and interpretability (Guo et al., 2020). A KG consists of facts commonly represented as triples of the form subject-predicate-object. It is a heterogeneous graph where nodes function as entities and edges represent relations between entities (Fensel et al., 2020). KGs provide a structured representation of data, capturing relationships and attributes among entities.

KG may develop with the content side perspective. It provides the mutual relations between items (Zhang F. et al., 2016). Those graphs are called item graphs. User-side information may also be integrated into the KG to make relationships between users and items more coherent. It captures user preferences more accurately (Zhang Y. et al., 2018). Integrating KG as side information into RSs has been actively studied. It makes user-item relations more coherent and accurately captures user preferences (Geng et al., 2022).

KG-based approaches play a significant role in generating explainability by leveraging the connectivity paths within the graph. These approaches often utilize methods, such as embedding-based, path-based, or hybrid methodologies, to enhance the interpretability of the information stored in the KG (Guo et al., 2020).

Though KGs assist in providing accurate recommendations and explanations, they can be difficult to traverse due to their complex graph structure. Embedding-

based or KGE models can help to address this challenge by representing entities and relations in a low-dimensional space, which makes it easier to find paths between them (Ji et al., 2022). One research direction involves utilizing KGE models, such as translational distance models (e.g., TransE (Bordes et al., 2013), TransH (Wang Z. et al., 2014), TransR (Lin et al., 2015), RotatE (Sun et al., 2019)) and semantic matching models (e.g., DistMult (Yang et al., 2015)), to ascertain the entity similarity in Euclidean and Complex-Vector space. Pure KGE methods leverage the semantic representations of entities and their relations embedded in vectorized forms but lack the path connectivity information among entities. Hence, there are limitations in discovering multi-hop relational paths, which restricts the model's ability to recommend entities and generate explainabilities that are more than one hop away (Ji et al., 2022).

Another line of research investigates path-based recommendations. Path-based methods focus on extracting meaningful paths or sequences of relationships between entities within the KG. These paths can provide insights into the logical connections between entities, contributing to explainability. By analysing paths, one can uncover chains of relationships that lead from one entity to another, shedding light on the reasons behind certain associations. For example, (Gao et al., 2018) proposed the notion of meta-paths to reason over KGs. However, the approach has difficulty coping with numerous relations and entities in large real-world KGs and cannot explore relationships between unconnected entities. Recently, (Xian et al., 2019) proposed Policy-Guided Path Reasoning (PGPR) to use RL to search for reasonable paths between user-item pairs. They formulated the recommendation problem as a Markov decision to find suitable user-item connections in the KG. They trained an agent on sample paths between users and items by carefully designing the path-searching algorithm, the transition strategy, terminal conditions, and RL rewards. In the prediction phase, PGPR generates recommendations for users with specific paths to interpret the reasoning process. However, the approach has difficulty in capturing user preferences.

Recently, a new trend of research has focused on unifying the embedding-based with the path-based methods to exploit information from both sides. Hybrid

approaches combine both embedding-based and path-based methodologies to harness the benefits of both techniques (Wang H. et al., 2018). For instance, a hybrid approach might use embeddings to capture entity and relationship semantics while incorporating path analysis to reveal specific chains of relationships that explain connections between entities. This integration allows for a more comprehensive and nuanced understanding of the KG's content. However, challenges persist with this approach, particularly regarding the refinement of user representation and the scalability of the models (Guo et al., 2020).

This research proposes a framework to generate XR and focuses on quantifying the explainability of the recommendations. This involves developing metrics or measures to assess how well the generated recommendations can be explained to users. By quantifying explainability, the researchers aim to provide a clear understanding of the effectiveness and transparency.

1.3 PROBLEM STATEMENT

Explainability holds paramount importance for both businesses and users, fostering a deeper understanding of AI recommendations and facilitating tailored customer service. The challenge persists due to the opacity of AI models, prompting the emergence of the XAI field. This field delves into techniques such as KGs, embeddings, RL, and DL to render AI models more transparent. Quantitative evaluation methods further enhance the quality of explainability, collectively advancing the comprehension and utility of AI recommendations (Adadi & Berrada, 2018) (Evans et al., 2022). The following sections delve deeply into the problem statement that this research addresses.

1.3.1 Enhancing Explainable Recommendations

RSs leverage the structured information within KGs to enhance recommendation accuracy and explainability. By tapping into the interconnected relationships between users, items, and attributes, KGs enable recommendations to consider both direct and indirect connections. This results in more precise suggestions, while the graphical representation of these connections fosters user understanding of the rationale behind

recommendations. This approach proves particularly beneficial in domains where relationships and context significantly influence user preferences, promoting better recommendation outcomes and user engagement (Wang H. et al., 2018a).

Path-based recommendations (e.g., PGPR (Xian et al., 2019), EKar (Song et al., 2019), ReMR (Wang et al., 2022), and KGDQN (Xu et al., 2021)) applying RL as a Markov decision process (MDP) utilizes the viable paths between user-item pairs. However, most RL approaches overlook rewarding agents for pursuing intermediate paths that are subsets of guided path patterns and may align with user preferences. Furthermore, finding meaningful traversal paths on large graphs remains challenging, impacting the overall accuracy of recommendations.

(Xian et al., 2019) proposed PGPR, combining MDP and RL to navigate multi-hop paths in the KG for offering recommendations, and thus, ensured transparency by providing connectivity paths, but faced challenges in computational costs, personalization, optimal product suggestions, and their associated reasoning. Notably, the approach prioritizes reasoning path selection solely based on path probability, overlooking other factors. In a subsequent study, (Xu et al., 2021) developed KGDQN by integrating KGs into a deep Q-network (DQN) for item recommendations. Yet, it inherits limitations from DQN, like overestimation bias and instability. While (Xu et al., 2021) introduced a non-binary rewards strategy, it still grapples with issues concerning optimal products and their associated reasoning paths, determined by maximizing the multiplication of rewards with corresponding probabilities. (Song et al., 2019) introduced EKAR, which employs deep RL and MDP to craft understandable paths based on user preferences. However, this approach demands an ample amount of personalized policy training data. Additionally, it lacks an intermediary rewarding mechanism and may get binary rewards of (+1, 0, or -1) depending on the path completion state with the terminal entity. It faces challenges with suggesting optimal recommended products and their corresponding reasoning, as it predominantly relies on path probabilities or rewards for prioritizing products. (Wang et al., 2022) proposed ReMR, a method leveraging ontology and instance KGs to model multi-level user interests, enhancing the modelling of user interests and leading to improved recommendations. (Zhang et al., 2022) introduced RKGR-RNS,

focusing on the reward mechanism in RL for XR and introducing a negative sampling method into the RL model to effectively delineate user preferences. (Song et al., 2023) proposed HR-RL-KG, a method that utilizes RL to enable agents to traverse diverse paths more effectively and leverage the underlying KG by employing the TransD embedding structure.

In summary, each model targets different aspects of explainability, user preference adaptation, and knowledge representation to suit varied application needs. However, challenges remain regarding personalization, optimal product recommendations, and identifying effective reasoning paths for these recommendations.

1.3.2 Explainable Recommendations – Quantitative Evaluation

Even if employing KG as side information can help explainability, the vital issue is how to quantify or evaluate that explainability. Researchers aiming to quantify the explainability of RSs, especially those utilizing KGs, employ various evaluation methods. These include user studies that gauge user perceptions, online evaluations conducted in real-world scenarios to observe user interactions, and offline evaluations using historical data to assess explanation quality (Vultureanu-Albişi & Bădică, 2021). (Wang et al., 2018) evaluated the usefulness of generated explainability of their proposed approach NARRE by recruiting four Yelp users who have written at least 20 Yelp reviews to label the randomly selected restaurant explanations generated through their model. Similarly, (Liu et al., 2020) conducted a crowdsourcing activity by choosing the top 100 most active users and evaluating the generated explanations from their proposed approach, Ante-RNN. These evaluations offer insights into how well explanations enhance user understanding and trust in recommendations, aiding the development of effective and transparent RSs.

The metric or measurement for the quality of explainability predominantly evaluated through qualitative approaches or user studies has a subjective or opinionated purview and largely depends on user feedback. Qualitative studies, a known basis for evaluating XAI, may suffer from confirmation biases (Lin et al., 2021). Thus, there is a need for quantitative evaluation metrics to measure the

effectiveness of the explanations to avoid human biases. (Rosenfeld, 2021), proposed a quantitative metric to measure explainability, but it still required human interaction to evaluate the performance difference between the agent model and the logic presented as an explanation.

While exploiting KGs into RL-based RSs can boost recommendation performance and explainability, evaluation metrics of explainability are still not mature enough. Most of the current works that proposed explainability metrics (Afsar et al., 2022) (Hailemariam et al., 2020) (Lin et al., 2021) are qualitatively measured and, thus, not integrated as part of the overall recommendations' performances. As a result, it is desirable to consider more robust and quantifiable metrics that can be integrated into the overall evaluation of recommendation performance.

Overall, the problem statement categorizes the development of an XRS with better performance metrics and an evaluation metric for quantifying those explainabilities. This research tackles these challenges by promoting user preference integration into routes and enhancing personalization. It optimizes knowledge representation and recommendation by identifying significant traversal paths. A quantitative framework evaluates generated explanations, prioritizing products for improved accuracy. These efforts advance personalization, optimal pathfinding, and recommendation accuracy.

1.4 RESEARCH QUESTIONS

Given the above problem statement, it can be concluded that RSs still face major problems related to providing explainability associated with the generated recommendations that could negatively affect the end-users trust and confidence in the RSs and hamper their performances. As a result, further research efforts are still needed to fill the gaps by developing enhanced XRS to resolve the existing shortcomings. Thus, to achieve that, this study proposed to answer the following main research question (RQs):

- **RQ1**: Does the proposed framework enhance **personalized recommendation** by **rewarding the agents for following the path patterns aligning with user preferences**?

- **RQ2**: Does the proposed framework enhance the **embedded explanation** of the RSs **by identifying the optimal explainability path that maximizes the explainability to the user for the product**?

- **RQ3**: Does the proposed framework enhance the **semantic recommendation** of the RSs by **prioritizing the products for the user based on their affections, novelties, and utilities, ultimately understanding the context and semantics (relations) in the KG**?

- **RQ4**: Does the proposed framework **quantitatively measure the explainability efficacies** by considering both **valuable options** and **novelty or diversity** of the generated explainability for the product recommendations?

1.5 RESEARCH HYPOTHESIS

Considering the above research questions, the associated hypothesis for this research work are as follows:

- **RH1**: The proposed framework enhances **personalized recommendation** by **rewarding the agents for following the path patterns aligning with user preferences**.

- **RH2**: The proposed framework enhances **embedded explanation** of the RSs **by identifying the optimal explainability path that maximizes the explainability to the user for the product**.

- **RH3**: The proposed framework enhances the **semantic recommendation** of the RSs by **prioritizing the products for the user based on their affections,**

novelties, and utilities, ultimately understanding the context and semantics (relations) in the KG.

- **RH4**: The proposed framework **quantitatively measures the explainability efficacies** by considering both **valuable options** and **novelty or diversity** of the generated explainability for the product recommendations.

1.6 RESEARCH AIMS AND OBJECTIVES

This research aims to enhance overall recommendation performance and provide a suitable explanation for the generated recommendations by making greater use of the KG, KGE, and RL. This research leverages the semantic representation of entities and relations and the connectivity information in the KG. It develops an RL framework encompassing KG to train an RL agent to learn a policy.

The objectives of the research are as follows:

- **RO1**: To propose a method that addresses the limitations of existing RL KG-based recommendation models by **rewarding the agents to adhere to path patterns that align with user preferences**.

- **RO2:** To enhance the **embedded explanation** of the RSs **by identifying the optimal explainability path that maximizes the explainability to the user for the product**.

- **RO3**: To enhance the **semantic recommendation** of the RSs by **prioritizing the products for the user based on their affections, novelties, and utilities**.

- **RO4**: To propose a **quantitative measure to evaluate the explainability efficacies** by considering both **valuable options** and **novelty or diversity** of the generated explainability for the product recommendations.

Table 1.1 provides a summary of the problem statements, research questions, and corresponding research objectives.

Table 1.1 List of Problem Statements, Research Questions and Corresponding Objectives

Problem Statements	Research Questions	Research Objectives
Existing XRS model targets different aspects of explainability, user preference adaptation, and knowledge representation to suit varied application needs. However, challenges remain regarding personalization, optimal product recommendations, and identifying effective reasoning paths for these recommendations.	1. Does the proposed framework enhance **personalized recommendation** by **rewarding the agents for following the path patterns aligning with user preferences**? 2. Does the proposed framework enhance the **embedded explanation** of the RSs **by identifying the optimal explainability path that maximizes the explainability to the user for the product**? 3. Does the proposed framework enhance the **semantic recommendation** of the RSs by **prioritizing the products for the user based on their affections, novelties, and utilities, ultimately understanding the context and semantics (relations) in the KG**?	1. To propose a method that addresses the limitations of existing RL and KG-based recommendation models by **rewarding the agents to adhere to path patterns that align with user preferences**. 2. To enhance the embedded explanation of the RSs by **identifying the optimal explainability path that maximizes the explainability to the user for the product**. 3. To enhance the **semantic recommendation** of the RSs by **prioritizing the products for the user based on their affections, novelties, and utilities**.

to be continued…

…continuation

While exploiting KGs into RL-based RSs can boost recommendation performance and explainability, evaluation metrics of explainability are still not mature enough. It is desirable to consider more robust and quantifiable metrics that can be integrated into the overall evaluation of recommendation performance.	4. Does the proposed framework **quantitatively measure the explainability efficacies** by considering both **valuable options** and **novelty or diversity** of the generated explainability for the product recommendations?	4. To propose a **quantitative measure to evaluate the explainability efficacies** by considering both **valuable options** and **novelty or diversity** of the generated explainability for the product recommendations.

1.7 RESEARCH SCOPE

The scope of this research can be summarized as follows: RSs research is a combination of multiple research areas. It can be classified as an information filtering area because it aims to alleviate the impact of the information overload problem. It can also be classified as an information retrieval area because it aims to find relevant matching items for the user's requirement in the form of search keywords. In addition, it can be classified as a big data area because it deals with a massive amount of data that is increased daily at a high rate.

This research project encompasses a comprehensive exploration of RSs, encompassing information filtering and retrieval elements to counter information overload while addressing the challenges of handling large-scale data. Notably, it emphasizes the crucial aspect of explainability within recommendations, aiming to quantify the quality of explanations provided to users. By proposing a quantitative metric for evaluating the effectiveness of generated explanations, the research enhances user trust and comprehension in RSs, thus aligning with the evolving

landscape of AI research that values accuracy and transparency in decision-making processes.

This research thoroughly investigates XRS, leveraging the wealth of information present in KGs through entities and their interconnected relationships. The study employs techniques such as embedding, which involves transforming KG data into vectors and utilizes RL. By applying these methodologies, the research aims to identify the most likely paths users could take within the intricate network of connections, thereby enhancing the system's ability to provide meaningful and understandable recommendations.

More specifically, this research is confined to the following scopes:

- The research builds upon the framework introduced by (Xian et al., 2019) in their work published in 2019. By leveraging and expanding upon the concepts, methodologies, or findings presented in Xian et al.'s study, this research seeks to advance the understanding and application of XRS. This extension suggests a continuation of the previous work, potentially introducing new insights, modifications, or enhancements to the original framework to address specific challenges or limitations within the context of XR.

- The study centres around an exploration of prior research efforts in the realm of XRS, with a specific emphasis on those constructed using KGs as foundational structures, consisting of entities interconnected through relations. It delves into techniques such as embedding, which involves transforming the KG into vectors, and harnesses RL to uncover pathways connecting different entities. By synthesizing and analysing these established methodologies, the study aims to contribute insights into the creation of effective and interpretable recommendations within the context of KG-based XRS.

- This study primarily concentrates on the "TransE" algorithm as the chosen method for KGE, while not encompassing an investigation into alternative variations of embedding algorithms. The research is limited to examining the specific characteristics, capabilities, and potential drawbacks of the "TransE" approach within the context of KG-based systems, without extending its scope to encompass a broader exploration of alternative algorithms for achieving similar embedding outcomes.

- In this study, RL techniques are employed to construct an Actor-Critic model. This model is designed to identify the most likely connectivity paths of linked users about the candidate products that are under consideration for recommendation. By utilizing the principles of RL, the research seeks to enhance the understanding of user-product associations within the context of the RS. The Actor-Critic model is harnessed as a mechanism to efficiently navigate the complex network of connections to provide insightful recommendations based on the inferred pathways.

- In this research endeavour, four commonly employed e-commerce datasets are selected for analysis. Specifically, the study involves utilizing the Amazon dataset as part of its research methodology. These datasets are likely chosen due to their relevance and prevalence in the e-commerce domain, serving as representative samples for the investigation into the proposed RS and explainability methodologies.

- The proposed XR model is subject to evaluation using four distinct evaluation metrics: Normalized Discounted Cumulative Gain (NDCG), Hit Rate (HR), Recall, and Precision. These metrics are employed as quantitative measures to assess the performance and effectiveness of the model's recommendations. NDCG captures the relevance and ranking of recommended items, HR quantifies the ability to suggest relevant items, Recall measures the model's capability to retrieve relevant items, and Precision gauges the accuracy of recommended items. By employing this

array of evaluation metrics, the research aims to provide a comprehensive and multifaceted assessment of the proposed XR model's performance.

- The experimental procedures in this study were conducted using the Python programming language, specifically version 3.9.16 in a 64-bit configuration. The experimentation took place within the Scientific Python Development Environment, PyCharm version 2022.3.2 Community Edition. Furthermore, the DL Framework Pytorch version 1.13.1 was employed as a key tool in implementing the proposed methods and techniques. This choice of programming environment and tools underscores the research's commitment to utilizing established and capable resources for conducting experiments and analyses related to the XRS model.

1.8 RESEARCH METHODOLOGY

The research methodology involves a comprehensive approach spanning six phases: a thorough theoretical review of available literatures on RSs, explainability, and evaluation metrics; the collection of pertinent data; the design and implementation of the RS; the recommendation phase; a rigorous assessment and analysis of experimental outcomes, including user studies to gauge interpretability; concluding with the presentation of research findings and avenues for future expansion. The overall research methodology is depicted in **Figure 1.2**.

Phase 1 - Theoretical Study: In the theoretical study phase, the focus is on understanding essential concepts in XRS. The goal is to identify research purposes, challenges, and gaps in the field through a literature review. This phase helps define the research problem, set objectives, formulate questions, and establish research boundaries.

Phase 2 - Data Collection: In the next phase of the study, the focus shifts to data preparation and preprocessing. Four real e-commerce datasets from Amazon are utilized, covering various product categories. These datasets contain six distinct entities and eight different relationship types, establishing connections between users,

products, categories, and brands. Data preprocessing involves extracting relevant entities and user-product interactions, with a filtering criterion applied to retain significant feature words, ensuring dataset quality and relevance for subsequent analysis and modelling.

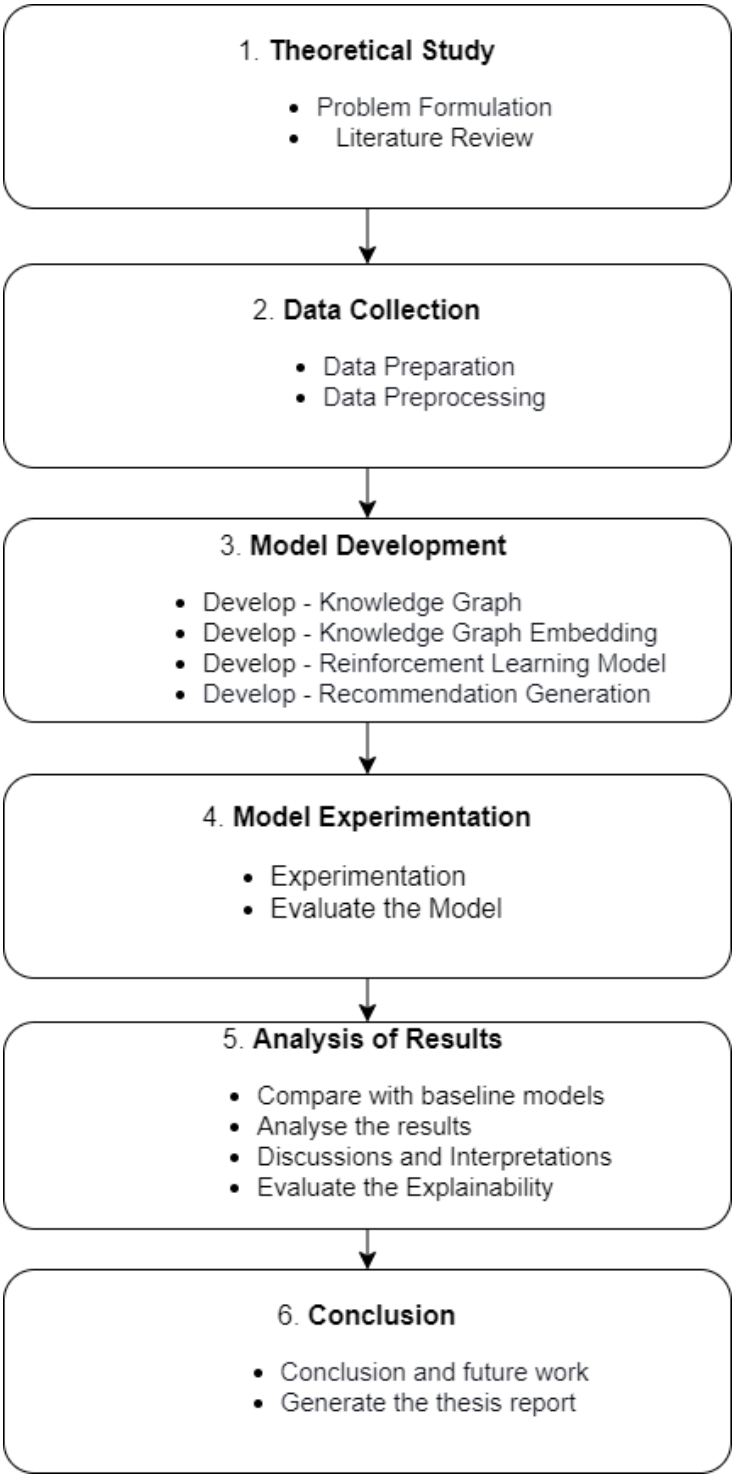

Figure 1.2 Research Methodology

Phase 3 - Model Development: In the third phase of the research, a comprehensive architecture is developed to address identified issues. It begins with creating a KG from data, representing users and products as entities and their interactions as relationships, in label property graph format. KGEs are then generated using the "TransE" algorithm to transform entities and relationships into vectors. Later, an Actor-critic RL model is designed and trained to navigate and understand connectivity pathways within the KG, closing the architectural blueprint. Later, advanced algorithms are employed for recommendation generation. These algorithms prioritize product recommendations based on their explainability and user preferences. A "max explainability score" metric is introduced to quantitatively assess recommendation transparency and comprehensibility, enhancing the quality of generated recommendations.

Phase 4 - Model Experimentation: In the next phase, the experimentation is conducted, and model's performance is rigorously assessed across all four datasets using important metrics such as NDCG, HR, Recall, and Precision. These metrics collectively evaluate the model's effectiveness in generating recommendations and provide insights into its strengths and areas for improvement.

Phase 5 – Analysis of Results: In this phase, the proposed model is compared with baseline models to identify its strengths and areas for improvement. It explores the model's explainability and discusses the implications of the results of the research objectives, providing a holistic understanding of the study's findings. Additionally, this phase conducts a case study to assess the impact of explainability.

Phase 6 - Conclusion: The culminating phase of the research includes a conclusion that summarizes key findings and contributions, aligning them with the original objectives. It is followed by a future work section that acknowledges limitations and suggests potential research directions for advancing the field of XAI, emphasizing advancement and progress.

1.9 SIGNIFICANCE OF THE RESEARCH

The significance of this research lies in its multi-faceted contributions to the field of RSs and explainability within the context of KG applications. By proposing and evaluating a novel XRS model XAIRec, the study addresses a critical need for transparent and interpretable recommendations in a world inundated with data. Besides, the research also proposes a metric for the quantitative evaluation of generated explainabilities and addresses a current research gap in RSs. The research's focus on KGs, KGE, RL, and evaluation metrics provides insights into the complex dynamics of user-product associations. Additionally, proposed algorithms and their chosen evaluation metrics add depth to understanding the algorithmic effectiveness and performance measurement. Ultimately, the research's potential to enhance user trust, refine recommendation quality, and pave the way for a transition towards XRS carries implications for improving user experience, promoting transparency in AI decision-making, and contributing to the broader discourse on trustworthy AI systems.

Overall, the significance of the research can be summarized as follows:

- **Addressing User Trust and Acceptance**: RSs have become integral to users' online experiences, influencing their choices and decisions. However, the "black box" nature of these systems often leaves users sceptical about the validity and fairness of the recommendations they receive. This research's focus on explainability directly tackles this issue by providing insights into the recommendations generation process, fostering trust, and increasing user acceptance of the system's suggestions.

- **Enhancing User Understanding**: The reason behind recommendations recommended by the RSs is crucial for users to understand their preferences, refine their choices, and build trust and satisfaction with the system. The proposed XRS model aims to bridge this gap by generating clear and coherent explanations for recommendations, helping users make informed choices, and educating them about the underlying data-driven decision-making process.

- **Advancing Knowledge Graph Applications**: KGs, with intricate webs of entities and relationships, have proven invaluable for capturing complex real-world connections. By leveraging KGE and RL, this research explores new ways of utilizing these structures to generate more contextually relevant semantic recommendations. This research contributes to the growing field of KG applications, extending their potential beyond traditional uses.

- **Max Explainability Score - A Quantitative Metric for Evaluating the Generated Explainability**: While exploiting KGs into RL-based RSs can boost recommendation performance and explainability, evaluation metrics of explainability are still not mature enough. Existing works leverage qualitative metrics to measure the goodness of generated explainabilities and may induce human biases. As a result, it is desirable to consider a quantitative metric to measure the quality of the generated explainability. This research introduces a novel algorithm to quantitatively measure the quality of explainability in RL-based RSs that leverage KGs. By addressing the limitations of existing qualitative metrics, this algorithm aims to provide a more objective and less biased evaluation of the generated explanations, ultimately enhancing recommendation performance and interpretability. This research paves the way for further refining this metric to the growing field of KG applications, enabling their expansion into new areas beyond their current scope.

- **Redefining Evaluation Metrics**: The utilization of multiple evaluation metrics (NDCG, HR, Recall, and Precision) goes beyond a one-dimensional assessment of the model's performance. This approach recognizes the multi-faceted nature of recommendation quality and effectiveness, fostering a more comprehensive understanding of how well the proposed model aligns with real-world user needs and preferences.

- **Guiding Future Research**: By extending and building upon the work of (Xian et al., 2019), this research contributes to the ongoing evolution of

XRS. Its findings and methodologies serve as a foundation for future researchers to further investigate, refine, and expand upon the concepts and techniques introduced in the study.

- **Shaping Responsible AI**: In the broader landscape of artificial intelligence, the research aligns with the growing movement toward responsible and ethical AI development. The focus on transparency, explainability, and user-centricity reflects a commitment to creating AI systems that are accountable and aligned with societal values.

In summary, this research's significance lies in its potential to transform the RSs existing operation process, fostering user trust, promoting transparency, and advancing KG-based methodologies. Its contributions resonate beyond the specific domain, offering insights and approaches that have implications for AI, user experience, and ethical technology development across various industries.

1.10 ORGANIZATION OF THE THESIS

This section describes the organization of the thesis. There are seven chapters in this thesis, which are arranged as follows:

Chapter I - Introduction: This chapter serves as an introductory overview of the research effort, offering insights into the proposed study. It furnishes a broad context for the study, outlining its general background. Additionally, the chapter addresses the specific problem being tackled, accompanied by the articulation of research questions, formulation of hypotheses, and delineation of objectives. The scope of the research is delineated, demarcating the boundaries within which the study operates. The research methodology employed for inquiry is expounded upon, elucidating the approach and techniques employed for data collection and analysis. Furthermore, the significance of the research is highlighted, emphasizing its potential contributions and its relevance in addressing contemporary challenges.

Chapter II - Literature Review: This chapter serves as a foundational guide to the research study, offering a comprehensive exploration of RSs, encompassing their

definition, phases, objectives, methodologies, and evaluation criteria. It further delves into vital concepts like XAI, KGs, and RL, subsequently leading to an in-depth investigation of the synergy between XR and quantitative assessment of their effectiveness. By elucidating the definitions, needs, objectives, methodologies, applications, challenges, and other advancements within these domains, the chapter not only bridges crucial knowledge gaps but also identifies the existing research landscape's limitations, thus paving the way for the research study's focus and contribution to this evolving field.

Chapter III - Research Methodology: This chapter presents the detailed methodology adopted in this study. It encompasses the generic framework of the research and the steps required to carry out the research systematically. This chapter discusses the steps involved in addressing research problems and answering these research questions to fulfil the research goal and objectives. The chapter addresses the different stages of the research, including research phases, methods, and tools.

Chapter IV – XAIRec Framework: This chapter serves to unveil the intricate workings of the proposed XAIRec framework, a significant advancement within the realm of RSs. Central to this advancement is the integration of XAI principles, which empowers the framework to bridge the gap between machine learning complexity and user comprehensibility. Through this fusion, XAIRec not only delivers precise product recommendations but also provides users with a deeper understanding of the underlying recommendation process. The chapter commences with a comprehensive overview of the framework, highlighting its pivotal role in achieving the objectives outlined in the initial chapter. By incorporating semantic technologies, XAIRec presents a novel approach to fulfilling the core research goals, emphasizing a user-centric perspective. Furthermore, the chapter delves into the framework's design, presenting a detailed exposition of the RUPEP, MES, and PPS algorithms. This elucidation unveils the underlying mechanics of these algorithms, bringing to light the advanced strategies that contribute to the framework's effectiveness. The chapter is dedicated to unraveling the pivotal aspect of generating explainability associated with the recommendations. It outlines how the framework provides insights into the decision-making process, ensuring that users gain clarity on why certain

recommendations are presented to them. Moreover, the chapter extends its coverage to the creation of a quantitative metric that measures the quality of explainability, adding a valuable dimension to the evaluation process.

Chapter V – Model Experimentation: The experimentation chapter in this research study plays a pivotal role by thoroughly describing the dataset used, the experimental setup, chosen evaluation metrics, the data splitting strategy, the research methodology for developing recommendation models, and the evaluation mechanism for assessing their performance. This chapter serves as the empirical foundation for the study, offering a comprehensive and transparent framework for testing and evaluating the proposed RS, ensuring a rigorous assessment of its performance and competitive edge within the research domain.

Chapter VI – Analysis of Results: In this chapter, the study embarks on a comprehensive exploration of the research, delving into the intricate details of the XR model, its constituent components, and the potentially transformative impact it can have within the realm of RSs. The journey is underpinned by a commitment to transparency and effectiveness, as the study meticulously evaluates this model throughout its development and evaluation phases. Additionally, the study rigorously compares it with state-of-the-art baseline models, offering in-depth analyses and illuminating examples that collectively contribute to the ongoing dialogue surrounding the transparency and efficacy of RSs.

Finally, Chapter VII - Conclusion: This chapter summarizes the research work conclusions discussed throughout this study. The chapter also highlights the significant contributions of the research and outlines potential directions for future studies.

CHAPTER II

LITERATURE REVIEW

2.1 INTRODUCTION

This chapter offers an extensive review of the literature that encompasses multiple facets of RSs. It begins by elucidating RSs' foundational underpinnings, expounding on their distinct phases, goals, approaches, and performance evaluation metrics. Various approaches such as CF, content-based filtering, and hybrid methods are explored, alongside an examination of performance metrics commonly employed to gauge the effectiveness of these systems.

Transitioning into the realm of explainability, the chapter underscores its paramount importance in the context of AI systems and provides an insightful discourse on the nuances of evaluating explainability. This evaluation is delineated through both qualitative avenues like user surveys and usability testing, as well as quantitative metrics that gauge the cohesiveness and lucidity of explanations.

This chapter offers an in-depth exploration of KGs and RL, elucidating their significance in the context of achieving the objectives set by XAI. The chapter establishes essential foundational knowledge underpinning subsequent discussions by delving into these concepts. This understanding serves as a crucial steppingstone, facilitating a smooth transition into the subsequent elaboration of research requirements and the clarification of XRS.

The chapter advances to elucidate the nascent concept of XRS, emphasizing their dual objectives of furnishing recommendations and rendering the decision-making process intelligible. This is underpinned by a thorough investigation into methodologies that facilitate explainability, notably including KGs, KGEs, and RL

techniques. The intersection of these techniques not only enriches the transparency of recommendations but also augments predictive accuracy.

The chapter culminates with an incisive survey of innovative works in the domain of XRS, thereby elucidating contemporary advancements and their role in synergistically advancing recommendation accuracy and interpretability.

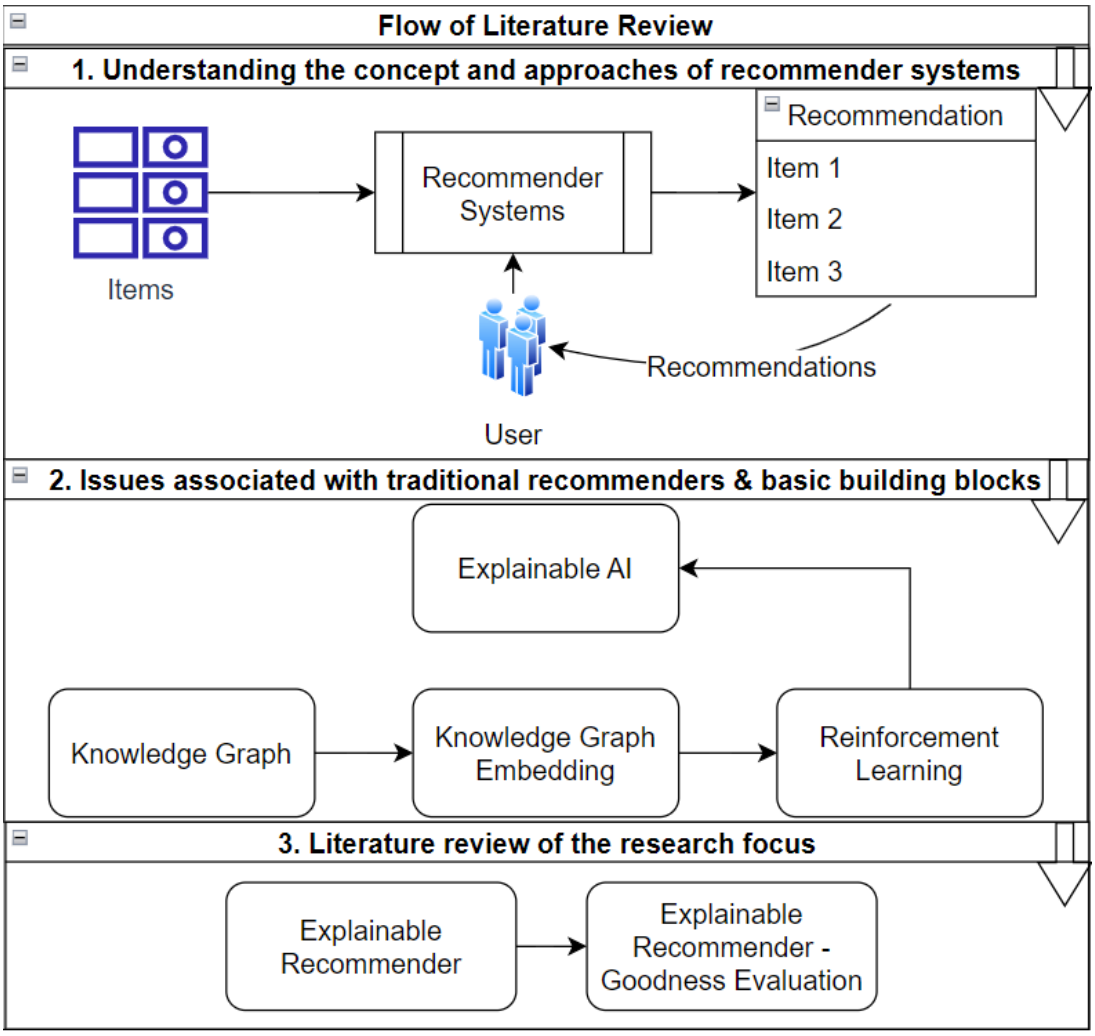

Figure 2.1 Flow of Literature Review

The chapter's structure unfolds as follows: Section 2.1 provides an introduction and flow of the literature review as depicted in **Figure 2.1**; Section 2.2 introduces

RSs, emphasizing traditional approaches; Section 2.3 explores explainability importance in AI solutions; Section 2.4 outlines some key notations; Section 2.5 delves into KG fundamentals; Section 2.6 covers KGE concepts and fundamentals; Section 2.7 explains RL and related literature; Section 2.8 elaborates on XRS; Section 2.9 discusses evaluation metrics for recommendation quality; Section 2.10 identifies research gaps; and Section 2.11 concludes the chapter.

2.2 RECOMMENDER SYSTEMS

RSs are integral to modern daily life, where people rely on recommendations to navigate an overwhelming array of choices (Resnick & Varian, 1997). These systems emulate social recommendation processes in digital settings, offering personalized suggestions based on user preferences and interactions. The term recommendation means recommending content to users and guiding them in their journey (Silveira et al., 2019). By doing so, they facilitate decision-making and fuel revenue growth and customer loyalty for online businesses (Kim et al., 2021). With applications spanning domains like movies (Huang et al., 2018), music (Hu et al., 2018), POIs (Xi et al., 2019), news (Wang H. et al., 2018a), and education (Huang et al., 2019), RSs employ diverse approaches such as collaborative and content-based filtering, hybrid techniques, and advanced algorithms to tailor recommendations to individual users, making their online experiences more efficient and satisfying.

RS is described comprehensively in the following subsections, including its definition, phases, goals, approaches, and performance evaluation metrics.

2.2.1 Overview

The RS has emerged as a pivotal tool in the digital landscape to address the challenge of information overload by serving as an effective information filtering mechanism (Resnick et al., 1994). The concept of information filtering encompasses two fundamental processes: "filtering in," which entails locating desired information for users, and "filtering out," aimed at excluding undesired information (Resnick et al., 1994). RSs have found applications across various platforms, including e-commerce and social networks, facilitating personalized recommendations (Lu et al., 2018). The

primary objective of RSs is to aid users and customers in discovering pertinent items that align with their requirements and, more importantly, their preferences.

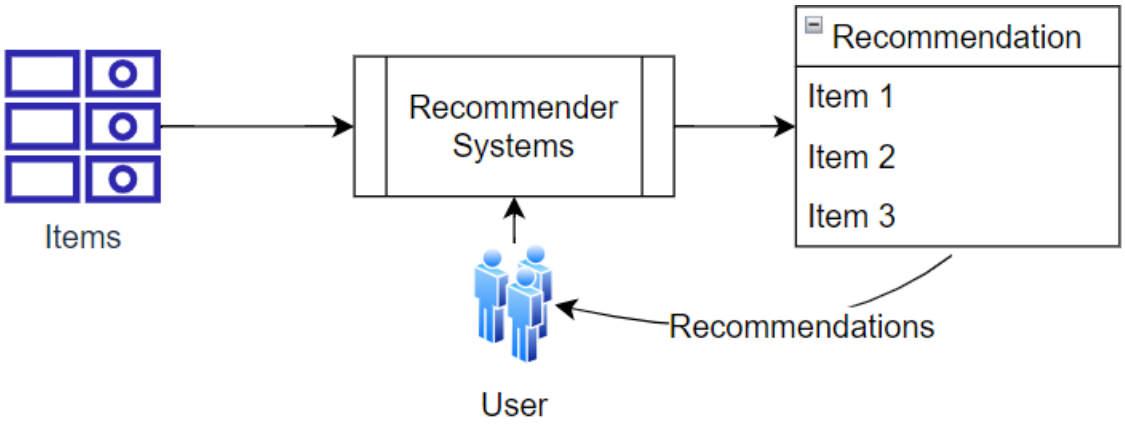

Figure 2.2 General Recommendation System Architecture

RSs operate by harmonizing user data, item data, and filtering techniques to deliver personalized suggestions. The user or customer data encompasses individual preferences, historical interactions, ratings, and reviews, furnishing valuable insights into their inclinations. Concurrently, item data encapsulates the attributes and features of distinct items, thereby facilitating a comprehensive understanding of the available choices. The filtering techniques, which encompass CF, content-based filtering, and hybrid approaches, draw upon the amalgamation of user and item data to discern patterns and relationships, generating tailored recommendations that resonate with users' interests and preferences. This triad of user data, item data, and filtering techniques synergizes to address the information overload challenge, culminating in an enhanced user experience. A general illustration of the RS architecture is depicted in **Figure 2.2**.

An RS model consists of a set of Users and Items and a utility function. All users are included in the Users set, and the Items set contains all the items that can be recommended to users. The utility function calculates the suitability of a recommendation to user $u \in Users$ and item $i \in Items$, which is declared as

$$R: Users \times Items \rightarrow R_0 \quad \ldots(2.1)$$

where R_0 is equal to either a real number or a positive integer within a specific range and represents the user's affection for the item. Finally, the recommendation can be generated by sorting item preference scores. (Adomavicius et al., 2011).

2.2.2 Goals of Recommender Systems

The primary goal of the RSs is to solve the problem of information overload due to the exponential growth flow of information on the Web to provide users/customers with different sources about services such as products, hotels, and restaurants (Ebadi & Krzyzak, 2016). To accomplish this primary goal, RSs generally consider several common objectives, such as relevance/accuracy, novelty, serendipity, and diversity. These four objectives are briefly described as follows:

- **Relevance/Accuracy**: Relevance and accuracy are synonymous and expressed as the prominent evaluation metrics for the RS. Users are more concerned with selecting items relevant to their interests. As a result, RSs focus on recommending items that are relevant to users' preferences. The RSs accuracy is determined based on the relevancy of the items to user needs and their liking by users (Shi et al., 2014). The high values of the performance metrics, such as precision, recall, and F-measure, and lower values for the error metrics, such as MAE and MSE, indicate better RS accuracy.

- **Novelty**: Novelty is a fundamental aspect of the success of recommendations and an essential metric of customer satisfaction. RS novelty is the ability of an RS to generate novel recommendations for users. The recommendation is considered novel if it satisfies three distinct characteristics: it is not previously known (unknown), it meets the needs or expectations (satisfactory), and it is different from other options (dissimilarity), as per (Zhang L. , 2013). Unknown refers to items that are not known to the user. Satisfactory relates to items that provide satisfaction to the user, whereas dissimilarity relates to items that are dissimilar to the items in the user profile. Novelty is contrary to popularity; the

more popular the recommended items are for the user, the less novel the items are for the RS performance.

- **Serendipity**: The most closely related serendipity concept is unexpectedness, involving the user's positive reaction to previously unknown serendipitous items. Serendipitous items denote unexpected, novel, and relevant items to a user (Kotkov et al., 2017). RSs that recommend serendipitous items to users will significantly improve sales and create trust relationships with the users. Recommending items with fortunate discoveries will encourage the user's curiosity to enhance the user experience with the system in place.

- **Diversity**: Diversity is the opposite of similarity. RSs that provide diverse recommendations solve the over-fitting problem and increase the user's experience with the RS. Diversity generally applies to a set of items that have to do with how different the items are compared to each other. In other words, it relates to the internal differences within parts of an experience. Diversity will usually ensure that users are not dissatisfied by continually always receiving the same recommendations (Lu et al., 2015).

2.2.3 Recommendation Process

The recommendation process has three main phases (Adomavicius et al., 2011) as shown in **Figure 2.3**. The phases are as follows:

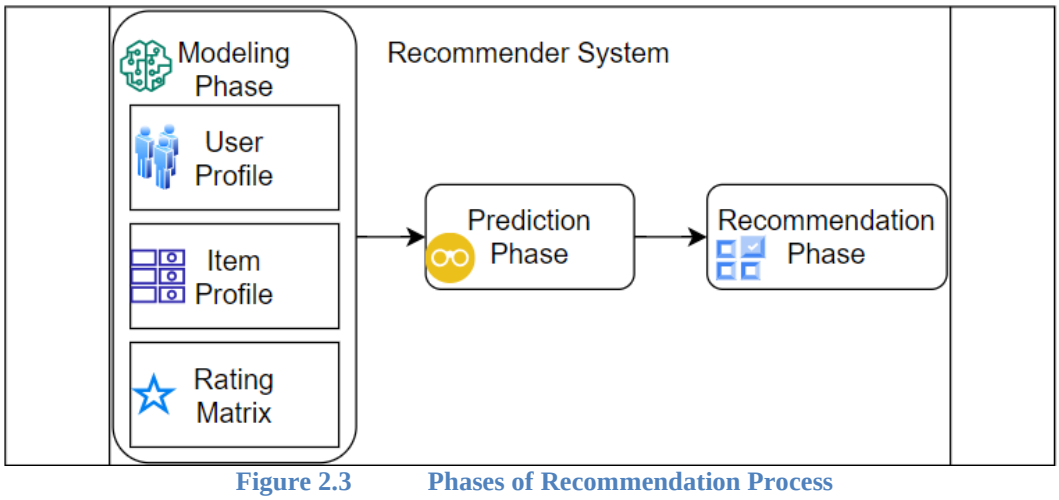

Figure 2.3 Phases of Recommendation Process

- **Modelling Phase**: This phase focuses on preparing the data that will be used in the subsequent two phases. There are three cases for that, and the first is building a user profile that is mostly a vector for each user that explains his preferences for an item as a whole or on few aspects of the item. Second, building an item profile containing a specific item's features. Third, creating a rating matrix that contains the users as records, items as attributes, and the value of each matrix's cell is the rating done by a user for a specific item.

- **Prediction Phase**: This phase aims to predict a specific user's rating or score of unseen/unknown items through a utility function depending on the extracted information during the modelling phase.

- **Recommendation Phase**: This phase is an extension of the prediction phase, where various approaches are applied to support the user's decision by filtering the most suitable items. It recommends/proposes items to the user (i.e., a set of top-N items with the highest-predicted ratings) that are attractive to him.

2.2.4 Implementation Approaches or Methodologies

Approaches to the recommendation are usually categorized into three main categories, as depicted in **Figure 2.4**: content-based, CF, and hybrid.

a. Content-based Filtering

Content-based recommendation implies that the items recommended to the user are similar to those the user liked previously. It assumes that users have interests in items similar to their past interacted items (Reddy et al., 2019). It generates the user profile from previously selected items by characterizing the user according to the item features and recommends items to the user based on the items with similar characteristics to the items that the user liked before (García-Cumbreras et al., 2013).

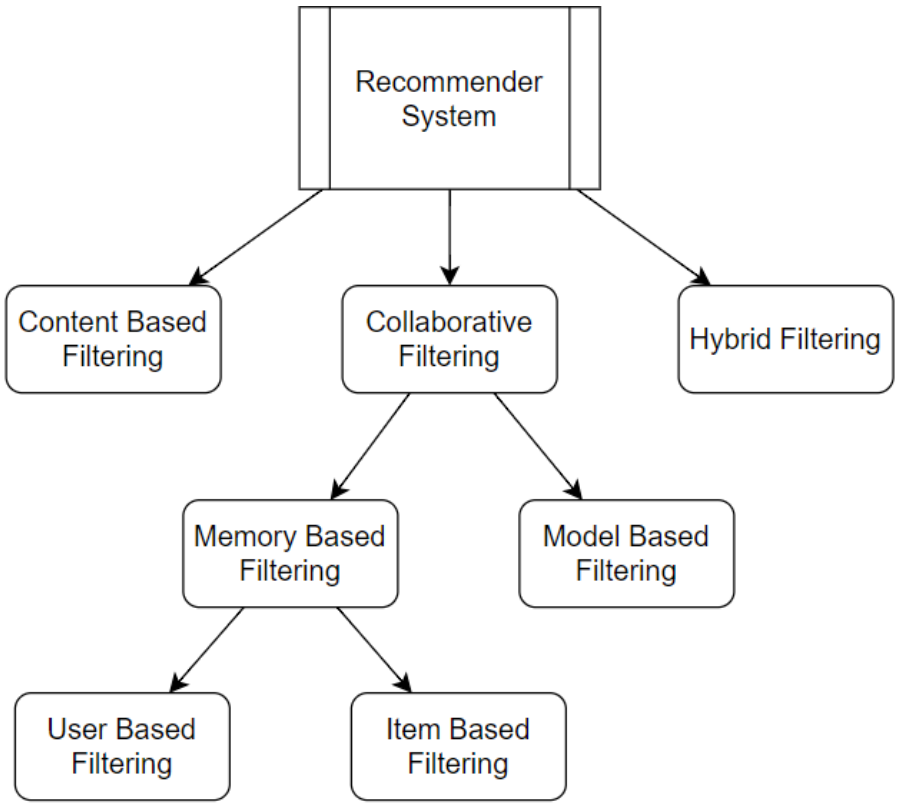

Figure 2.4 Recommendation System – Implementation Approaches

For example, a music RS may recommend songs with similar genres or artists that the user has shown interest in. These algorithms use various available content information such as the price, colour, brand, size, and much other information of the goods in e-commerce, or the genre, director, duration, casting, etc. of the movies in review systems to model RSs. It relies on the attributes of items and users to generate recommendations. Comparing candidate items with the user profile essentially involves matching them with their previous records. Therefore, this approach tends to recommend items similar to items liked by a user in the past, as depicted in **Figure 2.5**. Explaining the recommendation to users through various available item content information is intuitive, as they can easily understand it. Users may easily correlate with the suggested recommendations due to content features explanation (Wang et al., 2018).

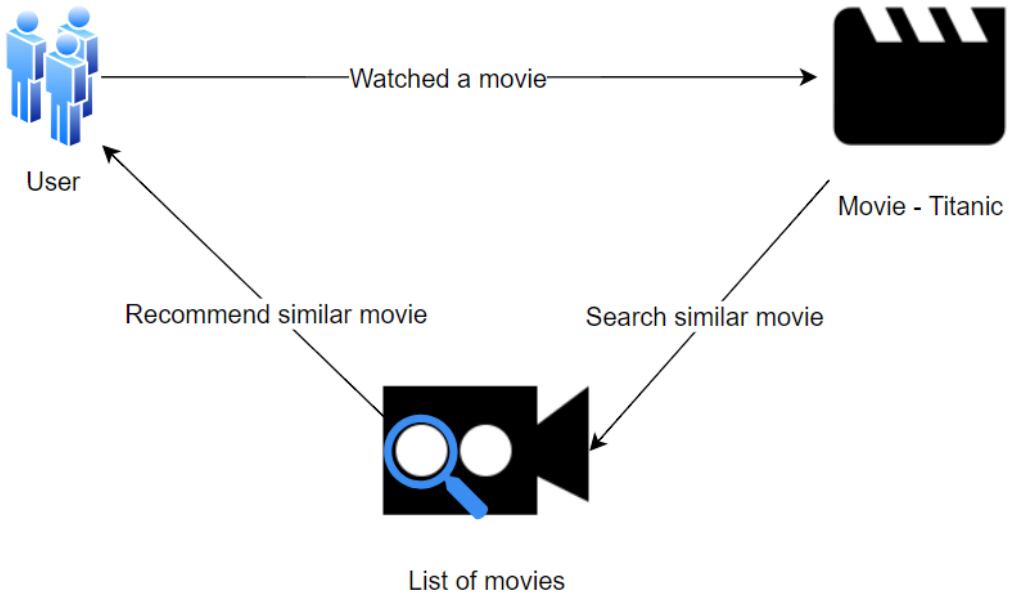

Figure 2.5 Recommendation System – Content-based Filtering - Illustration

i. Advantages of CB Approach

Some of the CB approach advantages are:

- It can explain how to recommend specific items (i.e., present the logic behind their recommendations) by providing a list of content features. This, in turn, can strengthen the user's confidence about the RS that reflects his preferences (Aciar et al., 2007).

- Since this approach relies on the content of each item, not the ratings of other users, it gives several advantages as follows (Pazzani & Billsus, 2007):

    - It offers a high level of personalization in the recommendations.

    - It is scalable in terms of the number of users.

    - It can make recommendations for users with peculiar interests.

- It has high security from malicious item creation and allows users to prevent viral marketing.

ii. Disadvantages of CB Approach

On the other hand, CB approaches have some disadvantages, such as:

- The vast size of the items is considered a major problem because when the recommendation is made, every item's content must be examined to discover items that are most likely relatable to the user's interest (Pazzani & Billsus, 2007). This task is error-prone and time-consuming (D'Addio & Manzato, 2014). Thus. they have the main demerit of collecting various content information in different application domains and are time-consuming.

- User profiles are built based on the static characteristics of the items. As a result, there is a high probability that different users have similar profiles even if they have various preferences among these items, just because they commented on the same items (Chen et al., 2015).

- The over-specialization problem occurs in this type of approach because users do not receive various or new items because of the restriction in their profile regarding the description of similar items (D'Addio & Manzato, 2014). Thus, these algorithms have the main demerit of not being able to consider recommendations for unexpected items (Marcuzzo et al., 2022).

b. Collaborative Filtering

The CF approach is the most popular technique used in RSs. It generates recommendations for a user based on the similarities among users with similar preferences/interests in the past (Al-Ghuribi & Noah., 2021). These algorithms are mainly using the wisdom of the crowds. The idea is that users who have similar preferences in the past are likely to have similar preferences in the future. This approach relies on user-item interaction data to identify user patterns and similarities. CF identifies the new user-item association by determining the relationships between

users and the interdependencies between items (Yang et al., 2016). It uses the implicit knowledge of a community of users on used items to identify their relationships with other users who have not used/seen those items within the community (García-Cumbreras et al., 2013). This can be represented as a user × items matrix in which each cell represents the user rating of a particular item. The interaction can be explicit such as ratings, or implicit such as clicks and views (Wan, 2018); (Wang et al., 2020).

The first CF framework for RS was developed by (Resnick et al., 1994) called GroupLens. It recommends articles to the Netnews clients using the rating server, named Better Bit Bureaus (BBB), which gathers users' ratings to predict other user's scores on articles based on the CF hypothesis, clients who agreed to the rating of articles in the past they will probably agree in the future.

A basic description of the CF approach for explaining the previous hypothesis is shown in **Figure 2.6**, where, based on the similar likings of users, the items are recommended to other users.

CF can be grouped into memory-based and model-based (Chen et al., 2015).

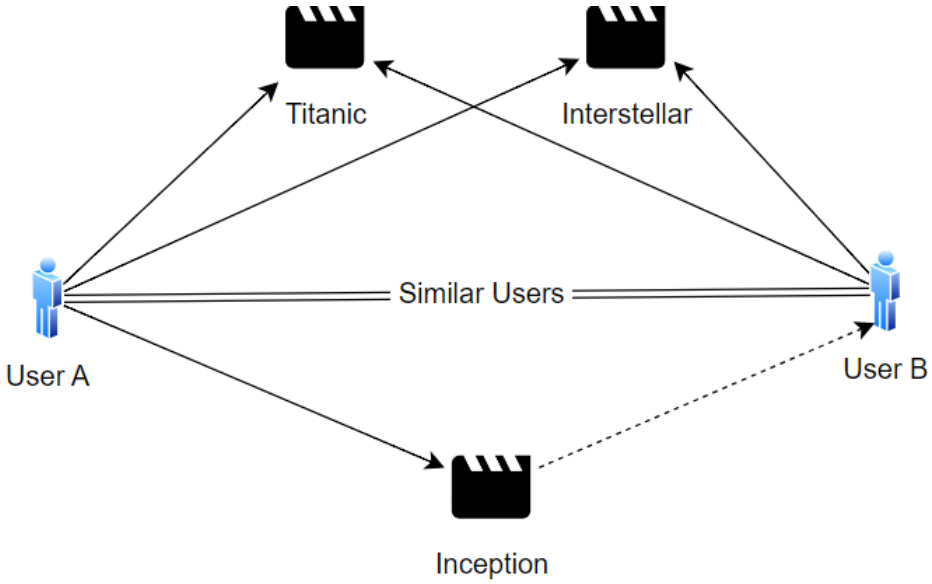

Figure 2.6 **Recommendation System – Collaborative Filtering - Illustration**

i. Memory-based Collaborative Filtering

The memory-based CF type is a heuristic algorithm that predicts the item's rating based on other users' ratings. It can be classified into two main sub-approaches. The first is user-based collaborative filtering (UBCF), and the other is item-based collaborative filtering (IBCF). The user-item rating matrix is the backbone of CF methods and predicts ratings via user-based or item-based CF methods. UBCF represents users as rating vectors and predicts missing ratings based on the weighted average from other similar users (Zhang et al., 2020). It identifies a set of neighbours (i.e., like-minded users) for a target user using ratings and then recommend a set of items that interest his neighbours. IBCF uses items as rating vectors and predicts missing ratings based on the weighted average ratings from similar items (Nawara & Kashef, 2021). It recommends items to a target user that are similar (i.e., has shared features) interests in the items that the user purchased, viewed, or liked before.

Figures 2.7 and **2.8** show a comparison between user-based and item-to-item CF recommendations. In the user-based approach, one can see that user 'User A' and 'User C' are similar because they liked common items. Thus, 'User C' is recommended to use the items that 'User A' likes, i.e., has a recommendation for movie "Inception". Instead, with the item-to-item approach, similarities between items are calculated based on which users liked them. One can see that 'Titanic' and 'A.I. Artificial Intelligence' are similar because they have been liked by users 'User A' and 'User B'. Thus, since 'User C' liked 'A.I. Artificial Intelligence', then 'Titanic' will be recommended to him.

The explainability power of CF lies in its design. For example, UBCF can explain the behaviour as "users that are similar to you, loved this item", and the IBCF as "the item is similar to your previously loved items". CF approaches have significantly improved the accuracy of the predictions. But it has less explainability power than content-based algorithms.

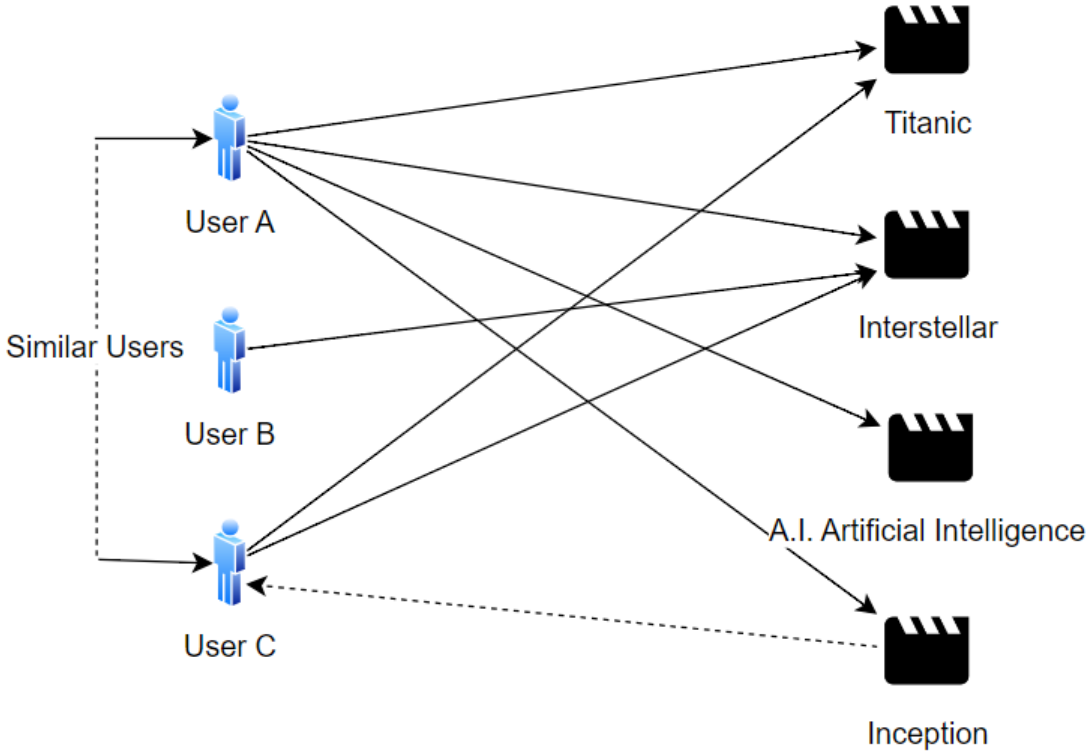

Figure 2.7 Recommendation System – UBCF – an Illustration.

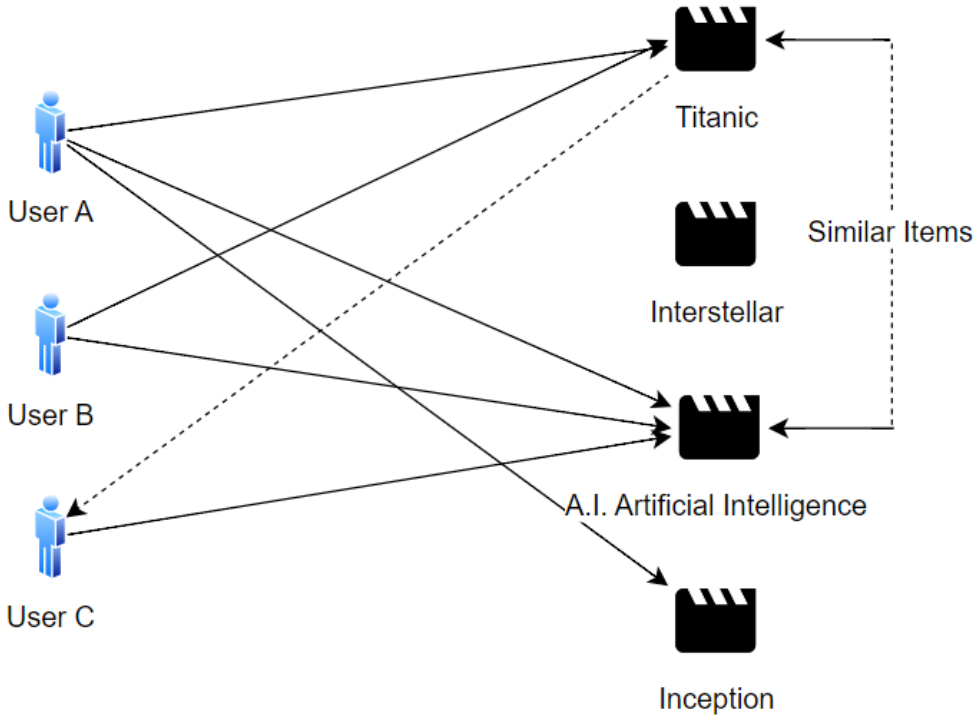

Figure 2.8 Recommendation System – IBCF – an Illustration.

ii. Model-based Collaborative Filtering

The model-based collaborative filtering approach attempts to alleviate the sparsity issue by building an inference model. One such implementation is the latent factor model (LFM) (Zhuang et al., 2017), which extracts the latent representation of users and items from the high-dimensional user-item interaction matrix, and then computes the similarity between users and items with the inner product or other methods. The most well-known LFM is matrix factorization (MF). LFMs, especially MF and its variants with CF (Koren et al., 2009), were very successful in rating prediction tasks and achieved great success. Other approaches in LFM are (SVD) singular value decomposition (Guan et al., 2017), (NMF) non-negative matrix factorization (Khalil et al., 2019), and (PMF) probabilistic matrix factorization (Yin et al., 2018). They create a latent factor representation to calculate the matching score of the user-item pairs.

However, latent factors in LFMs do not possess intuitive meanings and restrict the explanatory power of recommendations. This limitation restrains the users from understanding the reasoning for item predictions and reduces the trust and reliability of the system.

iii. Deep Learning

CF methods got further enhancements through DL. Similarity learning and representation learning are the two main approaches in DL methods of CF approaches. Similarity learning creates user-item embeddings and computes user-item similarity matching scores (Yang et al., 2020). The representation learning approach learns much richer user-item representations (Liu et al., 2020).

The accuracy of RSs has improved with the advent and application of DL-based technologies. DL-based recommendation engines have an issue explaining the reasons for recommending the recommendations to a user. Users must have explanations through model-generated RSs (Ai et al., 2018).

iv. Advantages and Disadvantages of CF Approach

CF based approaches have addressed the issue of content-based systems. They are the most fundamental and reliable approaches for recommendation engines. However, collaborative-filtering-based recommendation suffers from data sparsity and cold start problems (Marcuzzo et al., 2022; Sun et al., 2019).

A survey by (Su & Khoshgoftaar, 2009) provided a comparison between the CF classes as shown in **Table 2.1**.

Table 2.1 Comparison between CF Classes

CF Class	Advantages	Shortcomings
Memory-based	• Easy to implement. • Easy to add new data incrementally. • The content of recommended items need not be considered. • Co-rated items have been scaled well.	• Dependent on human ratings. • In sparse data, the recommendation performance is decreased. • New items/users cannot be recommended (i.e., cold start problem). • The large datasets have limited scalability.
Model-based	• The scalability and sparsity problems are better addressed. • The prediction performance is improved. • An intuitive rationale is given for recommendations.	• Building the model is expensive. • A trade-off is made between scalability and prediction performance. • Useful information is lost through dimensionality reduction techniques.

c. Hybrid Method

A hybrid method leverages content and CF techniques to overcome the limitation of using only one type of method. One major issue of CF-based recommendation is the sparsity of user-item interaction data, which limits finding similar items or users from

the interaction and leads to the cold-start problem. In this problem, the recommendation for a new user or an item is difficult as user-user and item-item similarity are not feasible due to no interaction records. One solution to address this problem is to incorporate content information of users and items into the CF-based framework (Sun et al., 2019). Hybrid RSs alleviate these issues by unifying the interaction-level similarity and content-level similarity to take advantage of their respective strengths and mitigate their weaknesses. Some commonly used item side information includes item attributes like brands and categories; item multimedia information, like textual description (Han et al., 2020), image features (Chu & Tsai, 2017), audio signals (Malleswari et al., 2023), video features (Daneshvar & Ravanmehr, 2022), and item reviews (Xu et al., 2021). Similarly, options for user-side information involve the user demographic information, including occupation, gender, hobbies, and user network (Wu et al., 2020). Overall, it allows using multiple kinds of side information, such as item attributes (Cano & Morisio, 2017), item reviews (Zheng et al., 2017), (Xu et al., 2018), and users' social networks (Chen et al., 2018). By combining different approaches, hybrid methods can provide more accurate and diverse recommendations.

2.2.5 Evaluation Metrics

As discussed in previous sections, RSs have different goals varying from the main to the sub-goals. A particular RS's success and efficiency often depend on the RS's primary objective and the domain characteristics to which it is applied (Schröder et al., 2011). Evaluation metrics for RSs can be broadly categorized into three (Herlocker et al., 2004): predictive accuracy metrics, classification accuracy metrics, and rank accuracy metrics.

a. Predictive Accuracy Metrics

Predictive accuracy or rating prediction metrics answer the question of how similar a recommender's predicted ratings are to actual user ratings. These metrics evaluate the RS performance by measuring the average error between the real ratings and the ratings predicted by the system. These metrics are mostly used as they are easy to

compute and understand and particularly useful for evaluating tasks in which the predicting rating will be displayed to the user (Herlocker et al., 2004).

As discussed in the works (Gunawardana et al., 2012); (Herlocker et al., 2004), there are three crucial prediction accuracy metrics used: Mean Absolute Error (MAE), Mean Square Error (MSE), and Root Mean Square Error (RMSE). Usually, these metrics calculate the error difference between the predicted ratings and the user's actual ratings; thus, a lower value implies a better RS performance for these metrics. **Table 2.2** defines the equations of the three metrics.

Table 2.2 Predictive Accuracy Metrics

Evaluation Metric	Details	Equation
MAE	• $\boldsymbol{N}$**:** is the size of the test set. • $\boldsymbol{p_i}$: predicted rating calculated by RS • $\boldsymbol{r_i}$: actual rating given by the user	$MAE = \frac{\sum_{i=1}^{N}(p_i - r_i)}{N}$
MSE		$MSE = \frac{\sum_{i=1}^{N}(p_i - r_i)^2}{N}$
RMSE		$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(p_i - r_i)^2}{N}}$

b. Classification Accuracy Metrics

Classification metrics determine the extent to which an RS makes correct or incorrect decisions about whether an item is good (i.e., good means relevant to user preferences) (Herlocker et al., 2004). Generally, a list of items for users is recommended by the RS. In general, they are organized horizontally or vertically, and users only care about the front parts of most items or the back parts of the items. A top-n recommendation is the name of this way of recommendation. The three most popular metrics are used to evaluate the RS recommendations' efficiency: precision, recall, and F-Measure.

The item set must be divided into two classes to calculate the previous metrics for RS ((Gunawardana et al., 2012); (Herlocker et al., 2004)): the first class determines the relevance of the item: relevant and irrelevant. The second class determines the quality of the returned items to the user selected/recommended and not selected/not recommended. The rating scale in specific datasets is not binary, so it needs to be converted into a binary scale. For example, the Amazon dataset's rating scale ranges from 1 to 5 to transform it to a binary scale; all the ratings of 1 to 3 are converted to irrelevant, and all the 4 and 5 ratings are converted to relevant. The item set classes are organized in **Table 2.3** to be used later in the metrics' formula.

Table 2.3 Class of Item Set

	Selected	**Not Selected**	**Total**
Relevant	N_{rs}	N_{rn}	N_r
Irrelevant	N_{is}	N_{in}	N_i
Total	N_s	N_n	$\boldsymbol{N}$

Based on the item set classes shown in **Table 2.3**, the description of each of the three famous classification accuracy metrics is shown in **Table 2.4**.

Table 2.4 Classification Accuracy Metrics

Evaluation Metric	**Definition**	**Equation**
Precision	The ratio of relevant items selected to the number of items selected	$P = \frac{N_{rs}}{N_s}$
Recall	The ratio of relevant items selected to total number of relevant items	$R = \frac{N_{rs}}{N_r}$
F-measure	The harmonic mean of Precision and Recall that offers a better measure of the incorrectly classified cases	$F - Score = \frac{2\,P\,R}{P + R}$

The equations shown in **Table 2.4** are the standard ones for the classification accuracy metrics. Some works, such as (D'Addio & Manzato, 2014) do measure these metrics for all the relevant items, a small k sample of the total ranking recommendations is only used as follows: precision@k, recall@k, and F-measure@k.

i. Precision@k

Precision quantifies the proportion of recommended items that are relevant to the user. It gauges the accuracy of the model's recommendations by assessing the ratio of relevant recommendations to the total number of recommendations. Higher Precision signifies a higher degree of accuracy in the recommendations provided. Precision measures the recommended items relevant up to a ranking position k.

$$P@k = \frac{1}{|u|}\sum_{u\,\epsilon\,u}\frac{|Rel_u@k|}{k} \quad \ldots(2.2)$$

Here Rel_urepresents the set of relevant items for user u, and $Rel_u@k$ is the number of relevant recommended items up to the position k.

ii. Recall@k

Recall is a metric that focuses on the fraction of relevant items that are successfully retrieved by the RS. In other words, it measures the ability of the system to find all the relevant items among the recommendations. A higher recall indicates that the system is effectively capturing a larger portion of the relevant items, which can be important for comprehensive recommendation coverage. Recall measures the relevant items recommended up to a ranking position k.

$$R@k = \frac{1}{|u|}\sum_{u\,\epsilon\,u}\frac{|Rel_u@k|}{|Rel_u|} \quad \ldots(2.3)$$

iii. F-Score@k

The F-score (F-measure) combines the Precision and the Recall score in a single value through the harmonic mean.

$$F@k = \frac{2*(P@k)*(R@k)}{(P@k+ R@k)} \quad \ldots(2.4)$$

c. Rank Accuracy Metrics

Rank accuracy metrics measure a recommendation algorithm's ability to create a recommended order of items that matches how the (Herlocker et al., 2004). Unlike the two previous accuracy metrics, this type does not seek to calculate the RS ability to accurately predict a single item's rating. Rank accuracy metrics are suitable for the RSs that provide the user with ranked recommendation lists and are concerned with differentiating between the "best" alternatives and just "good" alternatives of the recommended items. This type is suitable for the domains interested in ordering items and emphasizes the differences between the elements (Herlocker et al., 2004).

The most famous examples of the metrics in this category include Hit Ratio and Mean Reciprocal Rank (Chen et al., 2015). Hit Ratio at N (HR@n) is a metric for calculating how many "hits" are done in an n-sized list of ranked items. A "hit" can be described as an activity done by the user, such as ‘liked’, ‘purchased’, ‘clicked on’, or ‘saved/favourite’. HR@n measures the percentage of RS successes by calculating the average probability that the pushed item will be recommended by a top N recommender (Sarwar et al., 2001). Simply, HR@n shows whether the desired items appear on the recommendation lists (presence) and how many times they appear (frequency) (Dias & Fonseca, 2013). In contrast, Mean Reciprocal Rank (MRR) is a metric used to evaluate the generated recommendation lists to determine the ranking position of the target user’s choice in the recommendation list (Chen & Chen, 2014). The Reciprocal Rank (RR) calculates the reciprocal of the rank at which the first relevant item was retrieved. If the relevant item was retrieved at rank 1, the RR is 1, and 0.5 if the relevant document was retrieved at rank 2, and so on. When averaged across multiple queries, the measure is called the Mean Reciprocal Rank (MRR).

For example, Netflix suggests a list of movies that may also be interesting to the user, given the earlier watched movies. The objective is to predict the unforeseen movies the user might be willing to watch. These metrics include precision, recall, NDCG, mean average precision (MAP), and MRR. Each of these metrics captures the quality of a ranking from a slightly different angle.

i. Mean Average Precision (MAP)

The MAP metric provides a single summary of the user's ranking by averaging the precision figures obtained after the inclusion of each new relevant item, as follows.

$$MAP = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{1}{|Rel_u|}\sum_{i \in Rel_u} P@rank(u,i) \quad \ldots(2.5)$$

Where $rank(u,i)$ outputs the ranking position of the item i in the user u list; hence, precision is computed at the position where each relevant item has been recommended.

ii. Normalized Discounted Cumulative Gain (nDCG)

NDCG is a widely recognized metric that quantifies the ranking quality of recommendations. It accounts for both the relevance of recommended items and their positions in the list, providing a comprehensive measure of recommendation quality. A higher NDCG score indicates more accurate and relevant recommendations.

The Discounted Cumulative Gain (DCG) is an overall measure of the usefulness or gain of a list of retrieved items, weighted by the sorted list. This measure uses a logarithmic discount factor to give more weight to higher positions and penalize lower ones. The nDCG uses graded relevance accumulated starting at the top of the ranking and may discount at lower ranks.

$$nDCG@k = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{1}{IDCG_u^k}\sum_{i=1}^{k} \frac{(2^{Rel(u,p_i)}-1)}{log_2(i+1)} \quad \ldots(2.6)$$

Where the discount function is defined as $\frac{(2^{Rel(u,p_i)}-1)}{log_2(i+1)}$.

$IDCG_u^k$ denotes the score obtained by an ideal or perfect ranking of the user u up to the position k, which acts as a normalization factor to compare different users and datasets. $Rel(u,p_i)$ denotes the relevancy of an item p for the user u at a position i in the sorted list.

iii. Mean Reciprocal Rank (MRR)

Mean reciprocal rank (MRR) favours rankings whose first correct result occurs near the top-ranking results. It is defined as follows:

$$MRR = \frac{1}{|U|}\sum_{u \epsilon U}\frac{1}{rank(u)} \quad \ldots(2.7)$$

where $rank(u)$ is a function that returns the position of the first relevant item obtained for the user u.

iv. Hit Rate (HR)

The HR is the percentage of users with at least one correct recommendation. It is analogous to Recall. HR represents the frequency at which the model's recommendations match items that the user eventually interacts with. It reflects the model's ability to suggest items that align with user preferences. A higher HR implies that the model is effectively suggesting items that resonate with users.

2.2.6 Summary of the Section

Overall, this section covered a comprehensive overview of RSs, encompassing their architecture, objectives, development stages, implementation strategies, and performance assessment methods. Traditional algorithmic approaches like content-based and CF based on developing recommendation engines have merits and demerits. Notably, this section sheds light on the challenges confronted by conventional techniques, characterized by data scarcity and the resultant "cold start" hurdle arising from limited user engagements.

Furthermore, both traditional and advanced methodologies, including DL, exhibit an inherent black-box nature, lacking transparency in explicating recommendation rationales. The end users might lack visibility on the reasoning behind the recommended content. This opacity engenders user apprehensions concerning the reliability of these systems. In response, the emerging field of XAI

seeks to mitigate this transparency deficit, thereby enhancing user confidence and trust in recommendation mechanisms.

Also, there is no pre-defined best metric for evaluation, as each metric measures different aspects of the final ranking. Accuracy-based metrics consider only the distance between the actual and the predicted score. With ranking-based metrics, NDCG is comparatively better than other approaches at distinguishing between higher and lower-ranked items. To verify the trade-off between precision and recall for different values of k, MAP, and other measures can provide intuitive evaluation estimates. HR and MRR may also have utilities in a few situations. If it requires the predicted list to contain at least one relevant item (HR) or if the item must be present in the higher ranks (MRR).

2.3 EXPLAINABLE ARTIFICIAL INTELLIGENCE (XAI)

The evolution of AI and machine learning (ML) has advanced significantly from theoretical origins to becoming an essential element of contemporary technology-driven societies. While the rapid growth of AI/ML and integration into a wide array of civilian and military domains have yielded success, they have also introduced fresh complexities and barriers. Notably, the emergence of AI/ML systems capable of autonomous decision-making has underscored the importance of comprehending the decision processes. Consequently, a novel area of AI exploration, XAI, has emerged to address this need. The U.S. Defense Advanced Research Projects Agency (DARPA) defined XAI as “AI systems that can explain their rationale to a human user, characterize their strengths and weakness, and convey an understanding of how they will behave in the future” (Gunning & Aha, 2019).

Figure 2.9 provides a conceptual overview of XAI. Significantly, the concept of explainability holds a pivotal role in the establishment of reliable and transparent systems. In this context, the subsequent subsections delve into an exhaustive exploration of explainability, encompassing its precise definition, inherent significance and goals, targeted domains, strategies for implementation, metrics for evaluating performance, challenges, and the array of applications falling within this domain.

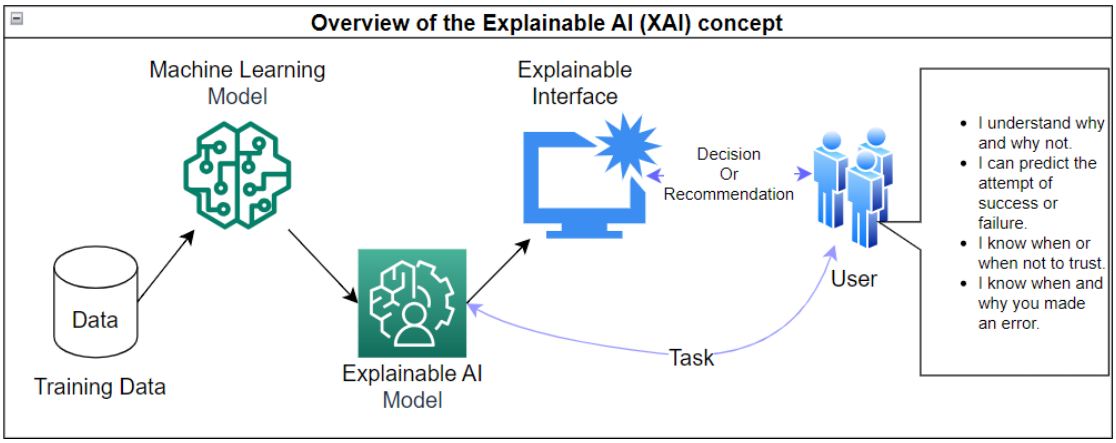

Figure 2.9 Overview of the eXplainable AI (XAI) Concepts

2.3.1 What is Explainability?

Explainability is a means to explain the logic behind the model intricacies, predictions, recommendations, and suggestions to end users and decision-makers. The goals of the explanation involve answering questions such as, “How does the AI system work?”, “How does it predict the specific recommendations?”, “Why were such recommendations generated for the user?”, “Which explanation for prediction or recommendation is most suited?” (Hoffman et al., 2018). Explainability makes it feasible for consumers to comprehend how the data is handled and alerts them to potential bias and system flaws. The ability to explain the model predictions bridges the confidence and trust of users in the system.

Figure 2.10 presents a conceptual model of XAI. It illustrates that user interaction with AI-generated outcomes initially forms a rudimentary mental model of both the task and AI system, often accompanied by initial mistrust. However, subsequent engagements involving system-generated explanations refine users' mental models, fostering improved task performance, trust, and reliance on the AI system.

By hypothesis, explanations that are both effective and satisfying for users have the potential to facilitate the development of a comprehensive and accurate mental framework. Consequently, this well-formed mental model would contribute to fostering a suitable level of trust in the AI system.

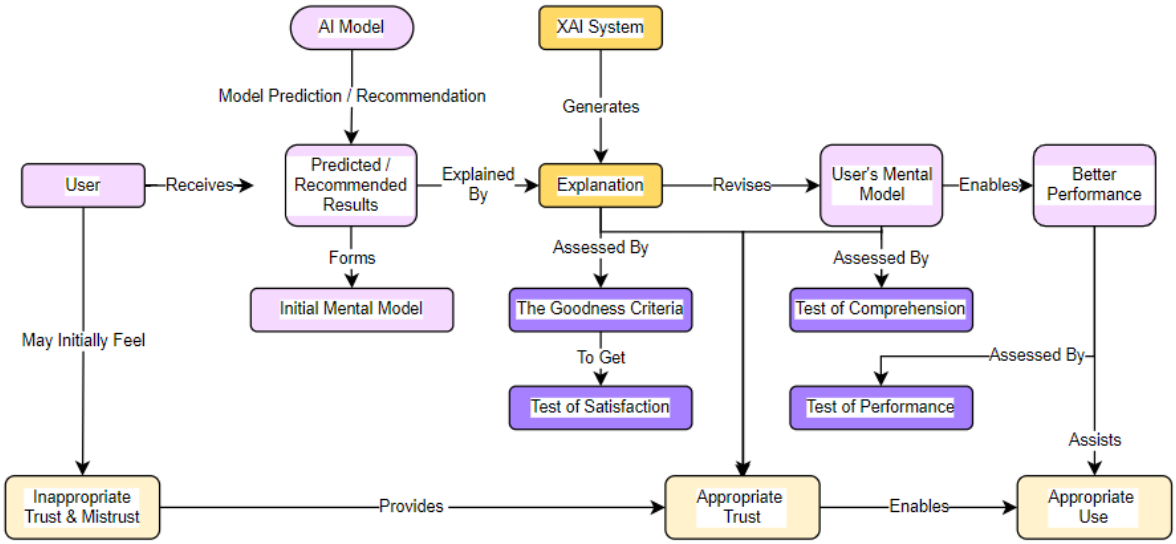

Figure 2.10 A Conceptual Model of the Explaining Process for the XAI

As a result of this established trust, users are expected to perform adeptly when interacting with the AI system. Thus, this process of better explainability not only enhances transparency but also contributes to users' evolving understanding, enabling them to make more informed decisions and effectively utilize the AI system's capabilities (Hoffman et al., 2018).

2.3.2 Need of Explainability

The enforcement of AI regulations such as the European Union GDPR (Voigt & Von dem Bussche, 2017), European Union AI Act 2023 (Laux et al., 2023), and the California Consumer Privacy Act of 2018 (de la Torre, 2018), underscores the "right to explanation" in algorithmic decision-making, highlighting the growing importance of AI system explainability. This emphasis arises from concerns about transparency, accountability, bias mitigation, and ethical considerations. All these developments lead to the importance of addressing the explainability issues of various IR/RSs.

The need for appropriate explanations led researchers gradually into XRS. The objective is to explain the recommendations predicted by the recommendation engine. (Zhang et al., 2014a) defined the explainability problems of recommendations and proposed an Explicit Factor Model (EFM) by aligning the latent dimensions with

explicit features. XAI is a salient topic for research in IR/RSs community. It has great applicability in search, conversational chatbots, and RSs.

Overall, the explainability of AI systems has always been an important topic. (Clancey, 1983) suggested the system requires far more knowledge to explain the predictions than just predicting them. The recent rise of big data, DL, computational powers, cloud storage, and other advanced technologies allowed AI performance to reach great heights. Researchers are now aligning with the importance of XAI in DL, computer vision, autonomous driving, natural language processing, video analytics, and many other advanced tasks.

2.3.3 Goals of XAI

The goals of XAI revolve around making AI systems more transparent, interpretable, and understandable to developers and end-users. As AI technologies become increasingly complex and integrated into various aspects of our lives, it becomes crucial to ensure that these systems can provide clear explanations for their decisions and actions. The primary goals of XAI include (Rawal et al., 2021):

- **Transparency:** XAI aims to create AI systems whose internal mechanisms and decision-making processes are not a "black box." This means that developers and users should have insights into how the AI arrives at its conclusions. Transparent AI systems enable easier identification of biases, errors, and potential issues (Bunn, 2020).

- **Interpretability:** AI models should produce results that are human-readable and comprehensible. Interpretability allows stakeholders to understand the mapping of inputs to outputs to make informed decisions (Gilpin et al., 2018).

- **Trust and Accountability:** Users and stakeholders can have more confidence in the AI systems if the system generates explanations for their decisions. This explanation is critical in applications like healthcare, finance, and autonomous

vehicles. XAI fosters accountability by making it possible to trace and understand the rationale behind specific decisions (Arrieta et al., 2020).

- **Fairness, Bias Detection and Mitigation:** Many AI systems can inadvertently learn biases present in the training data. XAI techniques can help identify and address these biases by providing insights into the data and features that influence the model's decisions (Voigt & Von dem Bussche, 2017).

- **Understandability:** End-users who might not have technical backgrounds should be able to comprehend the AI's actions. XAI makes it possible for users to understand why a recommendation is made, which can lead to more informed decisions (Mohseni et al., 2021).

- **Legal and Ethical Compliance:** As regulations related to AI and data privacy become more stringent, explainability can assist in demonstrating compliance with legal and ethical standards. Transparent AI systems are better equipped to adhere to regulations and explain their actions (Laux et al., 2023).

- **Causality:** Causality among data is an important source of information for XAI systems. Causality about explainability gave rise to a new term, causability. Causality, coined by (Holzinger et al., 2019), is defined as “the extent to which an explanation of a statement to a human expert achieves a specified level of causal understanding with effectiveness, efficiency, and satisfaction in a specified context of use”.

- **Privacy Protection:** Privacy and data protection are major challenges in many applications. Data-driven AI/ML systems must be designed with the privacy of information in mind, and XAI systems that use large datasets from the public domain must be able to protect consumers' privacy. This is a potential vulnerability, if an XAI system might reveal private data in the course of explanation. Therefore, XAI systems must be developed in such manner that can protect sensitive data (Arrieta et al., 2020) (Mohseni et al., 2021).

Overall, the goals of XAI are centred around enhancing transparency, accountability, trust, and usability of AI systems across various applications and domains. This helps bridge the gap between advanced AI technologies and those who interact with and rely on them.

2.3.4 Explainability Focus Areas

The XAI refers to two major explainability areas concerning the end user. The first is to provide an explanation of the function of the AI model, i.e., an explanation of the internal process methodology of the AI system. AI models are generally black box models. The decision-makers have no or less visibility over the functioning of these AI models. AI models based on the deep-learning concept have challenges in providing the internal model intricacies to decision-makers or end users. As part of explainability, the decision-makers or end users need to know the internal functioning of these black-box models (Adadi & Berrada, 2018).

The second is to generate the explainability for the recommendations/predictions generated by the AI systems. Besides model functioning, the decision-makers or end users also need to know the reasoning behind the model's prediction or recommendation. This reasoning will adequately justify the model's prediction or recommendation and generate the user's trust in the system. It also provides a methodology for user action improvement based on the feedback received from the model for the recommendation or prediction (Gunning et al., 2019).

2.3.5 Explainability Generation Approaches

XAI systems can be designed using either a transparent model approach or a post-hoc-explainability approach (Guidotti et al., 2018). The difference between these methods stems from systems that are inherently explainable by design and those that need to be made explainable. Post hoc explainability can be implemented after the system is complete to make it more explainable. This is done using post-development/design methods, such as text explanation, visual explanation, and local explanation (Vale et al., 2022). In contrast, embedded explainable systems produce inherent explainability from their design perspectives (Shi et al., 2022).

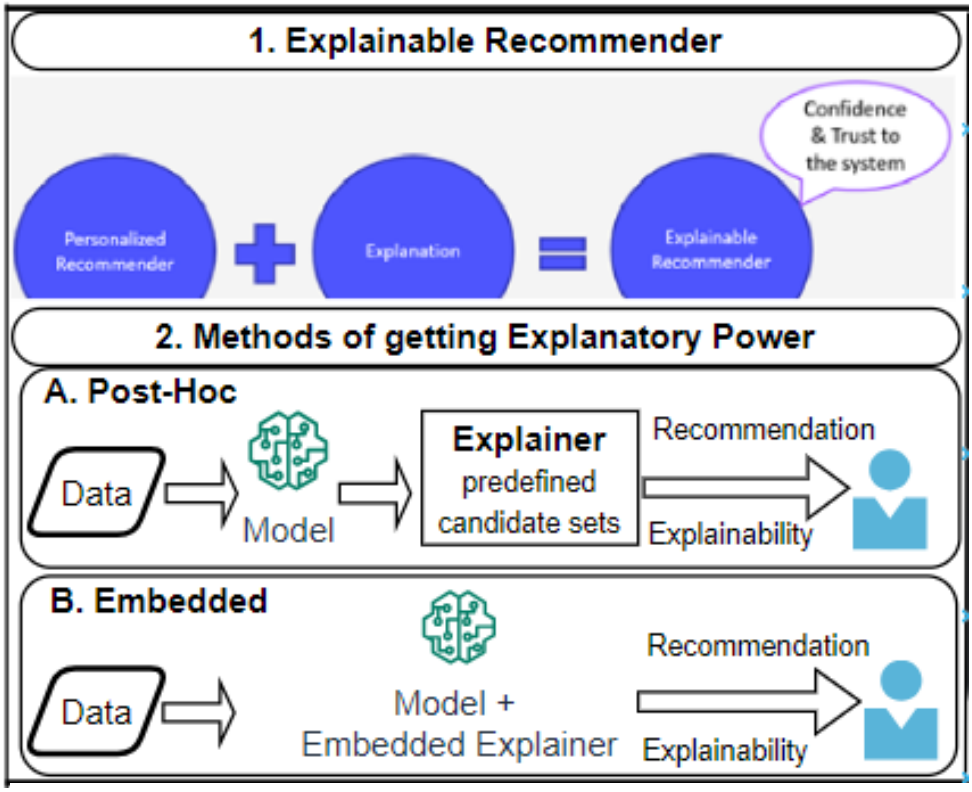

Figure 2.11 Explanation Generation – Approaches

The field of XR has gained significant attention and addresses the issue of transparency by providing users with understandable explanations for generated recommendations. Current XR approaches are categorised into post-hoc (model agnostic) and embedded (model intrinsic), as illustrated in **Figure 2.11**. Post-hoc methods generate explanations after the recommendation process, while embedded methods provide real-time explanations closely integrated with the recommendation process (Zhang & Chen, 2020) (Wang et al., 2018). KGs, with their semantic capabilities, enable the generation of model intrinsic or embedded explanations, which enhance the accuracy and transparency of RSs.

a. Post-hoc Methods

Post-hoc methods generate explanations after the recommendations. They identify reasoning from an earlier identified set of candidates, e.g., “people also bought” and “7 of your friends like this”. While the explanations are usually persuasive and highly readable, they neglect the working mechanism of the recommendation model. The

number of reasoning candidates limits the diversity of explanations. ECFKG (Ai et al., 2018) is an example of an RS that utilizes KGE technologies as its foundation.

b. Embedded Methods

Embedded methods incorporate the explanation process into the construction of a recommendation model. The explanations are text or images selected from the side information of the items. Usually, the pieces that contribute most to the accuracy of the recommendation are opted for. While the produced explanations are diversified and closely tied to the recommendation model, ensuring their readability and consistency is difficult.

Although post-hoc methods are model agnostic, they do not consider model explainability in generating diversified results (quality control). While the embedded methods have good model explainability, they are not model agnostic and are good at explanation quality control.

2.3.6 Effectiveness Evaluation of the Explainability

Different measures are needed to evaluate and verify the validity and performance of explanations given by XAI systems that may be designed with different explanation goals. To this end, DARPA's XAI program assessed XAI systems using the following measures (Gunning & Aha, 2019).

- **User Satisfaction:** User satisfaction measured the clarity and utility of the explanation based on the views of the end-user. Common approaches explored to measure usefulness, understandability/comprehensibility, and end-user satisfaction are user interviews (Gedikli et al., 2014), self-reporting questionnaires, Likert-scale questionnaires, and expert case studies.

- **Mental Models:** These are derived from the philosophical, psychological, and naturalistic models of human explanatory reasoning to measure the effectiveness of an explanation, i.e., the user's understanding of the system and the ability to predict its decisions in different situations (Rutjes et al., 2019).

- **Task Performance:** Task performance for the XAI system measures whether the explanation improves the user's decision-making or not, and also how well the user understands the XAI systems. User task performance was the evaluation of the user's performance for the designated task supported by the system. Studies from (Lim et al., 2009) investigated thend-users' performance, throughput, and prediction accuracy.

- **Trust Assessment:** For XAI systems, the user's trust in the system is a measure of its effectiveness. It is the evaluation of the user's ability to know when to trust or doubt the decisions made by the XAI systems. Studies by (Yin M. W., 2019) investigated the system's properties, such as accuracy, precision, inclusion, and level of explanation, affected users' trust in the system.

2.3.7 XAI Applications or Domains

The adoption of XAI can benefit various applications and domains by increasing transparency, accountability, and user trust in AI systems. Here are some areas that can benefit from the adoption of XAI:

- **Recommendation Systems:** To study how explanations help in RSs, (Tintarev & Masthoff, 2007) developed a prototype system to study the effect of different types of explanations, especially the relevant-user and relevant-item explanations. In particular, the author proposed seven benefits of providing recommendation explanations: transparency, scrutiny, trustworthiness, effectiveness, persuasiveness, efficiency, and satisfaction. Based on their user study, the authors showed that providing appropriate explanations can indeed benefit the RS over these seven perspectives. Recently (Vultureanu-Albişi & Bădică, 2021), shared a perspective of the application of XAI in RSs.

- **Healthcare**: XAI can help medical professionals, such as doctors and radiologists, understand the reasoning behind AI-assisted diagnostic decisions. This can lead to more accurate diagnoses and treatment plans, and doctors can better trust and collaborate with AI systems (Gerlings et al., 2022).

- **Finance**: In financial institutions, XAI can enhance risk assessment models, fraud detection, and algorithmic trading. Providing explanations for credit decisions or investment recommendations can help regulators, auditors, and clients understand the basis for these decisions (Ohana et al., 2021).

- **Autonomous Vehicles**: Self-driving cars need to make split-second decisions that impact safety. XAI can help both passengers and pedestrians understand why a self-driving car made a specific decision, especially in critical situations (Mankodiya et al., 2021).

- **Legal and Criminal Justice:** XAI can improve fairness and transparency in predictive policing and sentencing algorithms and is especially helpful in criminal justice and legal systems. This can help mitigate biases and ensure that decisions are made based on understandable criteria (Deeks, 2019) (Vale et al., 2022).

- **Industrial Processes**: In complex manufacturing and industrial processes, XAI can help engineers and operators understand why AI-controlled systems make certain choices. This can lead to more efficient processes and quicker troubleshooting (Kotriwala et al., 2021).

- **Education**: XAI can be applied to intelligent tutoring systems, helping students and teachers understand how the AI evaluates student performance and provides feedback. This transparency can lead to better-personalized learning experiences (Rachha & Seyam, 2023).

- **Climate, Environment, and Sustainability**: XAI can help researchers and policymakers understand the basis for predictions and recommendations made by AI models in climate modelling, environmental impact assessment, achieving sustainability through measuring greenhouse gas (GHG) emission, and natural disaster prediction (Zhang et al., 2023).

- **Cybersecurity**: XAI can enhance intrusion detection systems and threat analysis by providing insights into how AI detects and responds to various cyber threats (Srivastava et al., 2022).

In general, any application where AI-driven decisions impact human lives, safety, or well-being can benefit from the adoption of XAI. It helps build trust, uncover biases, prevent errors, and enables human-AI collaboration.

2.3.8 XAI – key Challenges

Adopting XAI can benefit various applications and domains by increasing transparency, accountability, and user trust in AI systems. However, XAI faces several key challenges that require addressing to ensure its effective implementation and adoption. Some of these challenges include (Arrieta et al., 2020):

- **Explainability versus Performance**: The trade-off for the balance of explainability and performance is also a major issue. As DL models become more and more complex and successful at solving learning problems, their inherent "non-transparency" presents a major challenge in making them explainable for XAI purposes. As stated by (Rudin, 2019), higher complexity does not inherently mean higher accuracy, and this has been very true for such DL models. As shown in **Figure 2.12**, a DARPA presentation provides a comparison of achieving higher performance versus the explainability for different ML algorithms. ML models with higher performance for prediction accuracy have lower explainability performance. It suggests a need for an optimal balance for both the system performance and explainability.

- **Lack of a Universal Standard**: One of the major challenges within the field of XAI is the terminology or ambiguity of definitions. As shown in the earlier sections, numerous terms are used when trying to articulate explainability to an AI/ML system. Furthermore, terms like interpretability, understandability, and comprehensibility have been used as synonyms, and only in the past few years have these terms taken on distinct meanings. However, a lack of a

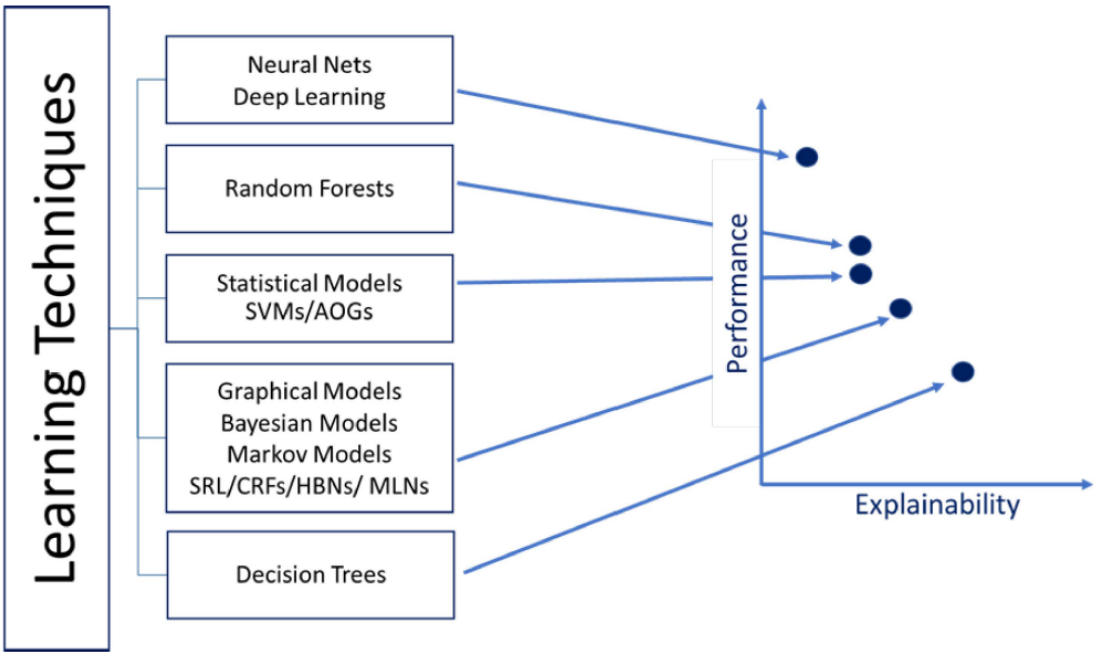

Figure 2.12 Comparison of ML Techniques, Explainability, and Performance (as per DARPA).

standard unified definition for the theory of explainability is noted. A unifying framework will provide common ground for researchers to contribute toward the properly defined needs and challenges of the field (Rawal et al., 2021).

- **Lack of a Quantitative Metric for Explainability Evaluation**: Quantitative metrics, other than simple interviews and questionnaires, are needed for measuring and evaluating the effectiveness of XAI. A study by (Hoffman et al., 2018) presented one of the only evaluation metrics for measuring the explanations of AI/ML systems. To this end, survey articles, such as the ones by (Arrieta et al., 2020) (Mohseni et al., 2021), will aid the overall development of XAI as an emerging new field.

- **Fairness of AI**: Another major concern for XAI coincides with one of the vital goals for the creation of such explainable systems is fairness and bias detection issues associated with the traditional AI/ML solutions. As the fields of accountable AI and XAI were born out of a need for fair and unbiased decision-making that affects human lives, getting rid of such biases remains a challenge within this young field. (Benjamins et al., 2019) noted that the discipline of fairness in AI inherently includes bias detection.

2.3.9 Summary of the Section

The preceding section has provided an all-encompassing overview of XAI, covering its definition, goals, focus areas, methodologies for implementation and evaluation, diverse applications, and the challenges it faces. Its objectives are to enhance transparency, accountability, user trust, and ethical AI deployment. XAI focuses on methods and techniques that make complex AI models interpretable, allowing users to grasp the reasoning behind their outputs.

XAI finds applications across diverse domains, including healthcare, finance, autonomous vehicles, legal and criminal justice, industrial processes, education, climate modelling, and cybersecurity. XAI contributes to better decision-making, fairness, and collaboration between humans and AI systems across these applications.

However, XAI faces significant challenges. These challenges include dealing with the complexity of models, striking a balance between accuracy and explainability, lack of standardization, ensuring human understanding, scalability of techniques, adapting to dynamic models, providing contextual explanations, handling causality and privacy concerns, addressing cultural variations, and navigating legal and ethical implications.

In the subsequent sections, the study will delve into key technologies pivotal in achieving the objectives of XAI and explore their integration into RSs. This exploration is valuable as RSs can greatly benefit from transparency and explanations that users can readily comprehend.

2.4 ENTROPY IN INFORMATION THEORY

The core idea of information theory is that the “informational value” or entropy of a communicated message depends on the degree to which the content of the message is surprising. If a highly likely event occurs, the message carries very little information. On the other hand, if a highly unlikely event occurs, the message is much more informative. For instance, the knowledge that some particular number will not be the winning number of a lottery provides very little information because any particular

chosen number will almost certainly not win. However, the knowledge that a particular number will win a lottery has high informational value because it communicates the outcome of a very low-probability event (Witten, 2020).

The information content or information value of entropy, also called the surprisal or self-information, of an event E is a function, which increases as the probability $p(E)$ of that event decreases. When $p(E)$ is close to 1, the surprisal of the event is low, but if $p(E)$ is close to 0, the surprisal of the event is high. This relationship is described by the function.

$$\mathbb{H}(E) = -log_2(p(E)) \quad \text{Or equivalently,} \quad \mathbb{H}(E) = log_2(\frac{1}{p(E)}) \qquad \ldots(2.8)$$

2.5 KNOWLEDGE GRAPH

Although the idea of an intelligent model encoding real-world “things, not strings” and their (inter-)relationships was already present in the literature since the ‘80s (Hogan et al., 2021), the term KG has regained popularity since Google announced their KG in 2012. Recently, multiple KGs have been proposed, such as Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015), YAGO (Suchanek et al., 2007), and Google’s KG (Singhal, 2012), which makes it convenient to build KGs for recommendation. Yet, there does not seem to be a precise definition of the term KGo date (Bonatti et al., 2019). However, there is a general understanding of its key characteristics.

A KG can be understood as a structured representation of information that captures relationships between entities in the real world. It's typically represented as a directed graph where nodes represent entities, and edges represent relationships between those entities. These relationships are often labelled to provide context or additional information (Hogan et al., 2021).

The structure of a KG can also include an ontological schema, which organizes entities and relationships into a hierarchical or categorical structure. This schema

helps to provide a more organized and structured view of the information within the graph (Yahya et al., 2021).

In the context of a KG, there are two essential components:

- **Terminology Box (TBox)**: This refers to the set of concepts, categories, and properties that define the vocabulary used in the KG. It establishes the foundational framework for understanding the entities and relationships within the graph (Fensel et al., 2020).

- **Assertion Box (ABox):** This includes the set of statements or assertions about individual entities, specifying their attributes, relationships, and other relevant information. The ABox contains specific data instances that fit into the concepts and relationships defined in the TBox (Fensel et al., 2020).

A KG aims to provide a more semantically rich and interconnected representation of data compared to traditional databases or flat data structures. It's used to enhance search engines, facilitate machine reasoning, support question-answering systems, and more by enabling a deeper understanding of the relationships between various entities and concepts in the real world.

In the subsequent subsections, this study will embark on a comprehensive exploration of the KG concept. This journey will encompass a detailed definition and an in-depth understanding of KGs, supplemented by illustrative examples, and pertinent concepts within this domain.

2.5.1 Examples of KG

Below are two examples shared in the entertainment and e-commerce industries. These examples illustrate the functioning of KG and its assistance in achieving the objectives of XAI. These examples revolve around the same person $Adam$ and explain the functioning of RSs for providing different kinds of recommendations to him based on his needs and requirements.

a. Example 1: Personalized Movie Recommendations in the Entertainment Industry

Figure 2.13 is an example of a KG-based movie recommendation in the entertainment industry where the movies "*Avatar*" and "*Inception*" are recommended to the user *Adam*. *Adam* is a movie enthusiast who loves exploring films across various genres. The entertainment platform leverages a KG and a content-based filtering RS to assist Adam in discovering movies he might enjoy. This KG contains entities such as users, movies, actors, directors, and genres, whereas interaction, belonging, acting, directing, and friendship are relations between entities.

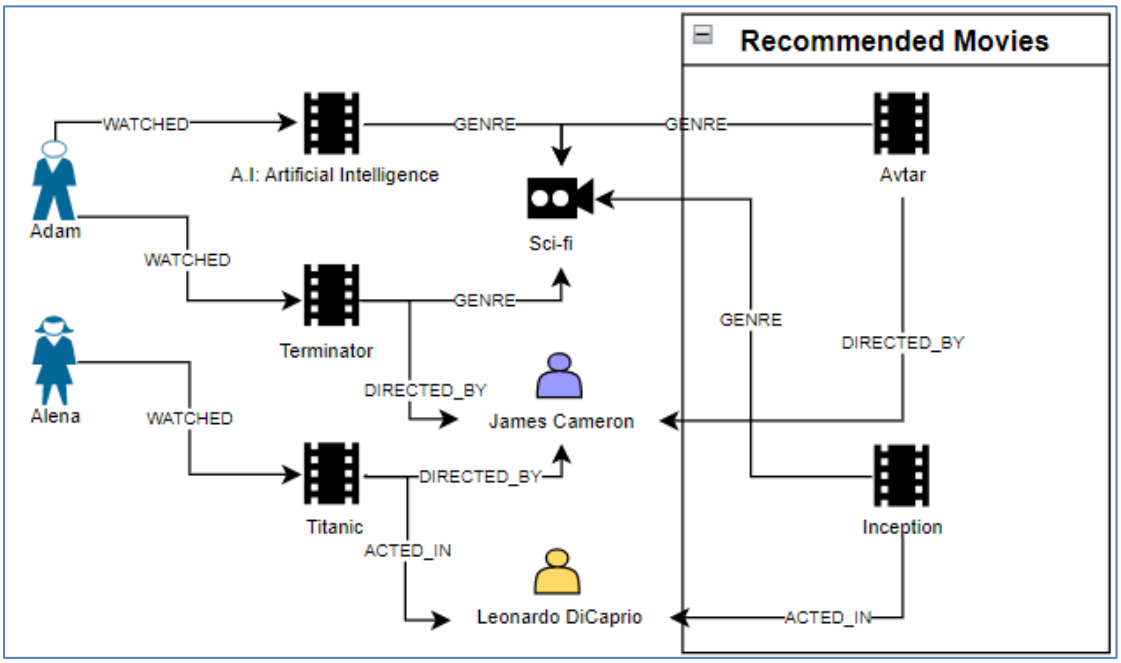

Figure 2.13 An illustration of KG-based movie recommendation.

The latent relationships between the movies and users may improve the recommendation precision. Another benefit of the KG-based RS is the explainability of recommendation results (Geng et al., 2022). Reasons for recommending these two movies to *Adam* become aware by following the related sequences in the user-item graph. For instance, one reason to recommend "*Avatar*" is that it falls within the same genre as "*A.I. Artificial Intelligence*", a movie previously watched by *Adam*.

b. Example 2: Personalized Product Recommendations in the e-commerce Industry

Adam is an avid online shopper, and he frequently explores various products on an e-commerce platform. The platform uses a KG and a CF RS to offer him personalized product recommendations. The KG in this e-commerce platform contains extensive information about products, brands, categories, customer reviews, and purchase histories. It captures the relationships between products and identifies patterns in users' preferences. The example depicted in **Figure 2.14**, illustrates the challenge of identifying the optimal traversal path in a KG for recommending a candidate item to a user in the e-commerce industry.

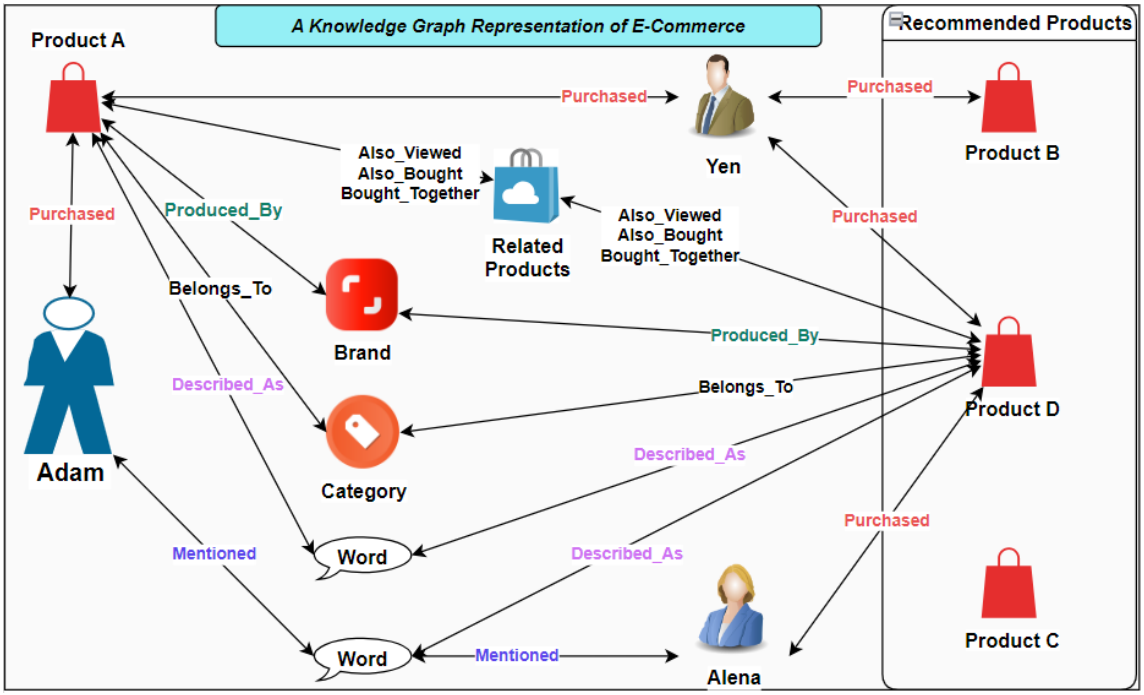

Figure 2.14 An Illustration of KG-based Product Recommendation in e-commerce

The objective of the recommendation model is to recommend a candidate item, such as $Product\ D$, to a user named $Adam$ with the optimal reasoning path in the graph, e.g., $\{Adam \xrightarrow{Purchase} Product\ A \xleftarrow{Purchase} Yen \xrightarrow{Purchase} Product\ D\}$ or $\{Adam \xrightarrow{Mention} Word\ A \xleftarrow{Described\_As} Product\ D\}$ or through other traversal paths, will provide explanatory reasoning for the recommendations given to the user. However, determining the optimal traversal path from numerous possibilities is quite challenging.

By utilizing a KG and various RSs, both the entertainment and e-commerce industries can offer personalized and transparent recommendations to users like $Adam$. These examples demonstrate how XAI becomes achievable using KGs, enabling users to comprehend the reasons behind the suggested content or products and increasing their engagement with the platforms.

2.5.2 Heterogeneous Information Network (HIN)

A Heterogeneous Information Network (HIN) is a directed graph $G = (V, E)$. Here V is a set of vertices or entities, and E is a set of edges or relationships (Guo et al., 2020). The entities may have different entity types. A HIN graph may contain data from various heterogeneous domains.

As depicted in the **Figure 2.13,** $Adam$ and $Alena$ are user entities, whereas $A.I. Artificial\ Intelligence$, $Terminator$, $Inception$, $Avtar$, and $Titanic$ are movie entities. On the other hand, $WATCHED$, $GENRE$, $DIRECTED\_BY$, and $ACTED\_IN$ are diverse relationship types in that directed graph.

Similarly, as depicted in the **Figure 2.14,** $Adam$, Yen, and $Alena$ are user entities; $Product\ A$, $Product\ B$, $Product\ C$, and $Product\ D$, are product entities; $Word$, $Brand$, $Category$, and $Related\ Products$ are other kinds of entities. On the other hand, $PURCHASED$, $MENTIONED$, $BELONGS\_TO$, $DESCRIBED\_AS$, $PRODUCED\_BY$, $ALSO\_BOUGHT$, $ALSO\_VIEWED$ and $BOUGHT\_TOGETHER$ are diverse relationship types in that directed graph.

2.5.3 Knowledge Graph (KG)

The KG is an example of a HIN. A KG $\mathcal{G}\text{know} = (V, E)$ is a directed HIN graph with V vertices or nodes and E edges or relationships. Here, nodes resembled entities, and edges are the subject-predicate-object triplet facts. Each edge of the form $< e_h(head\ entity),\ r(relation),\ e_t(tail\ entity) >$ indicates a relationship r, from the head entity e_h to the tail entity e_t (Guo et al., 2020).

As depicted in **Figure 2.13**, $< Adam,\ WATCHED,\ Terminator >$ is one such triplet that explains the fact that the user $Adam$ has watched the movie $Terminator$.

Similarly, as depicted in **Figure 2.14**, $< Adam,\ PURCHASED,\ Product\ A >$ is one such triplet that explains the fact that the user $Adam$ has purchased a product named $Product\ A$.

2.5.4 Meta-Path

A meta-path $\mathcal{P} = V_0 \overset{R_1}{\rightarrow} V_1 \overset{R_2}{\rightarrow} \cdots \overset{R_k}{\rightarrow} V_k$ is a path defined on the KG, $\mathcal{G}\text{know} = (V, E)$. The meta-path $\mathcal{P}$ comprises composite relations $R_1, R_2, \cdots R_k$ between entities V_0 and V_k. It is a relation sequence connecting entity pairs in the KG (Guo et al., 2020).

As depicted in the **Figure 2.13**, $< Adam,\ WATCHED,\ Terminator,\ DIRECTED\_BY,\ James\ Cameron,\ DIRECTED\_BY,\ Avatar >$ is one such meta-path.

Similarly, as depicted in the **Figure 2.14**, $< Adam,\ PURCHASED,\ Product\ A,\ PURCHASED,\ Yen,\ PURCHASED,\ Product\ D >$ is one such meta-path.

2.5.5 Meta-Graph

Similar to the theme of connection of a meta path, a meta-graph on the KG $\mathcal{G}\text{know} = (V, E)$ is a meta-structure connecting two entities or nodes. The meta-structure in the meta-graph defines as a combination of different meta-paths (Fang et al., 2016). A meta-path connects two entities with a sequence of relationships in a KG. So, due to multiple relation sequences, the meta-graph can provide more expressive structural information between entities in the KG.

$$\mathcal{P}_{MG} = \begin{cases} \mathcal{P}_{1V_{0k}}, \ where\ \mathcal{P}_{1V_{0k}} = V_0 \xrightarrow{R_{11}} V_1 \xrightarrow{R_{12}} \cdots \xrightarrow{R_{1k}} V_k \\ \mathcal{P}_{2V_{0k}}, \ where\ \mathcal{P}_{2V_{0k}} = V_0 \xrightarrow{R_{21}} V_1 \xrightarrow{R_{22}} \cdots \xrightarrow{R_{2k}} V_k \\ \quad \cdots \\ \quad \cdots \\ \mathcal{P}_{nV_{0k}}, \ where\ \mathcal{P}_{nV_{0k}} = V_0 \xrightarrow{R_{n1}} V_1 \xrightarrow{R_{n2}} \cdots \xrightarrow{R_{nk}} V_k \end{cases} \quad \ldots(2.9)$$

$$\mathcal{P}_{MG} = \{\mathcal{P}_{1V_{0k}}, \mathcal{P}_{2V_{0k}}, \cdots, \mathcal{P}_{nV_{0k}}\}$$

For example, as depicted in **Figure 2.13**, the meta-graph $\mathcal{P}_{MG}$ between the user $Adam$ and the movie, $Avtar$ is represented as:

$$\mathcal{P}_{1V_{Adam_{Avtar}}} = < Adam, WATCHED, A.I.\ Artificial\ Intelligence, GENRE, Sci-Fi, GENRE, Avtar >$$

$$\mathcal{P}_{2V_{Adam_{Avtar}}} = < Adam, WATCHED, Terminator, GENRE, Sci-Fi, GENRE, Avtar >$$

$$\mathcal{P}_{3V_{Adam_{Avtar}}} = < Adam, WATCHED, Terminator, DIRECTED\_BY, James\ Cameron, DIRECTED\_BY, Avtar >$$

$$\mathcal{P}_{MG} = \{\mathcal{P}_{1V_{Adam\_Avtar}}, \mathcal{P}_{2V_{Adam\_Avtar}}, \mathcal{P}_{3V_{Adam\_Avtar}}\}$$

Similarly, as depicted in **Figure 2.14,** the meta graph $\mathcal{P}_{MG}$ between the user $Adam$ and the product $Product\ D$ is represented as:

$$\mathcal{P}_{1V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, PURCHASED, Yen, PURCHASED, Product\ D >$$

$$\mathcal{P}_{2V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, PRODUCED\_BY, Brand, PRODUCED\_BY, Product\ D >$$

$$\mathcal{P}_{3V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, BELONGS\_TO, Category, BELONGS\_TO, Product\ D >$$

$$\mathcal{P}_{4V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, DESCRIBED\_AS, Word, DESCRIBED\_AS, Product\ D >$$

$$\mathcal{P}_{5V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_VIEWED, RelatedProducts, ALSO\_VIEWED, Product\ D >$$

$$\mathcal{P}_{6V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_VIEWED, RelatedProducts, ALSO\_BOUGHT, Product\ D >$$

$$\mathcal{P}_{7V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_VIEWED, RelatedProducts, BOUGHT\_TOGETHER, Product\ D >$$

$$\mathcal{P}_{8V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_BOUGHT, RelatedProducts, ALSO\_VIEWED, Product\ D >$$

$$\mathcal{P}_{9V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_BOUGHT, RelatedProducts, ALSO\_BOUGHT, Product\ D >$$

$$\mathcal{P}_{10V_{Adam_{ProductD}}} =$$
$$< Adam, PURCHASED, Product\ A, ALSO\_BOUGHT, RelatedProducts, BOUGHT\_TOGETHER, Product\ D >$$

$$\mathcal{P}_{11V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, BOUGHT\_TOGETHER, RelatedProducts, ALSO\_VIEWED, Product\ D >$$

$$\mathcal{P}_{12V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, BOUGHT\_TOGETHER, RelatedProducts, ALSO\_BOUGHT, Product\ D >$$

$$\mathcal{P}_{13V_{Adam_{ProductD}}} = < Adam, PURCHASED, Product\ A, BOUGHT\_TOGETHER, RelatedProducts, BOUGHT\_TOGETHER, Product\ D >$$

$$\mathcal{P}_{14V_{Adam_{ProductD}}} = < Adam, MENTIONED, Word, DESCRIBED\_AS, Product\ D >$$

$$\mathcal{P}_{15V_{Adam_{ProductD}}} = < Adam, MENTIONED, Word, MENTIONED, Alena, PURCHASED, Product\ D >$$

$$\mathcal{P}_{MG} = \{\mathcal{P}_{1V_{Adam_{ProductD}}}, \mathcal{P}_{2V_{Adam_{ProductD}}}, \mathcal{P}_{3V_{Adam_{ProductD}}}, \mathcal{P}_{4V_{Adam_{ProductD}}}, \mathcal{P}_{5V_{Adam_{ProductD}}}, \mathcal{P}_{6V_{Adam_{ProductD}}}, \mathcal{P}_{7V_{Adam_{ProductD}}}, \mathcal{P}_{8V_{Adam_{ProductD}}}, \mathcal{P}_{9V_{Adam_{ProductD}}}, \mathcal{P}_{10V_{Adam_{ProductD}}}, \mathcal{P}_{11V_{Adam_{ProductD}}}, \mathcal{P}_{12V_{Adam_{ProductD}}}, \mathcal{P}_{13V_{Adam_{ProductD}}}, \mathcal{P}_{14V_{Adam_{ProductD}}}, \mathcal{P}_{15V_{Adam_{ProductD}}}\}$$

2.5.6 User Feedback

Users may have implicit or explicit interactions with the items. These interactions could be in the form of implicit actions, such as users' clicking, watching, and browsing the items, or explicit actions, such as providing direct ratings. These interactions are termed user feedback and are defined as follows:

With m users $\mathcal{U} = \{u_1, \cdots, u_m\}$ and n items $\mathcal{V} = \{v_1, \cdots, v_n\}$, the binary user feedback matrix $I \in \mathbb{R}^{m \times n}$, defined as follows:

$$I_{ij} = \begin{cases} 1, & if\ (u_i, v_j)\ interaction\ is\ observed \\ 0, & otherwise \end{cases} \quad \ldots(2.10)$$

Note that a value of 1 for I_{ij} indicates the presence of an interaction between the user u_i and the item v_j. (Guo et al., 2020).

2.5.7 H-hop Neighbour

The KG $\mathcal{G}$know is a HIN between entities and relationships. Nodes in the KG may have connected with a multi-hop relation path: $e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_H} e_H$. A 1-hop means the target node is at a single relation distance from the source node. In this case, e_H is the H-hop neighbour of e_0, which can be represented as $e_H \in \mathcal{N}_{e_0}^H$. Note that $\mathcal{N}_{e_0}^0$ is e_0 itself (Guo et al., 2020).

For example, as depicted in **Figure 2.13**, the 2-hop neighbour of the user $Adam$ are:

$< Adam, WATCHED, A.I.\ Artificial\ Intelligence, GENRE, Sci-Fi >$

$< Adam, WATCHED, Terminator, GENRE, Sci-Fi >$

$< Adam, WATCHED, Terminator, DIRECTED\_BY, James\ Cameron >$

Here, $Sci-Fi$ is a 2-hop neighbour of $Adam$. Similarly, $James\ Cameron$ is a 2-hop neighbour of $Adam$.

On a similar note, as depicted in **Figure 2.14**, the 2-hop neighbour of the user $Adam$ are:

$< Adam, PURCHASED, Product\ A, PURCHASED, Yen >$

$< Adam, PURCHASED, Product\ A, PRODUCED\_BY, Brand >$

$< Adam, PURCHASED, Product\ A, BELONGS\_TO, Category >$

$< Adam, PURCHASED, Product\ A, DESCRIBED\_AS, Word >$

$< Adam, PURCHASED, Product\ A, ALSO\_BOUGHT, RelatedProduct >$

$< Adam, PURCHASED, Product\ A, ALSO\_VIEWED, RelatedProduct >$

$< Adam, PURCHASED, Product\ A, BOUGHT\_TOGETHER, RelatedProduct >$

$< Adam, MENTIONED, Word, MENTIONED, Alena >$

$< Adam, MENTIONED, Word, DESCRIBED\_AS, Product\ D >$

Here, uses *Yen*, and *Alena* are 2-hop neighbours of *Adam*. Similarly, *Brand*, *Category*, *Word*, and *RelatedProduct* are 2-hop neighbours of *Adam*. Even with some path, "*Product D*" is also a 2-hop neighbour of *Adam*.

2.5.8 RS Implementation with Advances in KG

Traditional algorithmic approaches like content-based and CF-based on developing recommendation engines have their merits/demerits. The main problem with the earlier traditional methods was the data sparsity issue. Very few items usually get the attention of users. Due to a lack of user interactions, platforms have no information about the degree of user preferences for all the available items. Users usually click or purchase very few products and provide ratings for very few of them. There are many items with no user interactions at all. It leads to the problem of a cold start. The KG has the potential to alleviate the issue of cold start. The KG provides rich and complementary information to user-item interactions by exploring the interlinks within a KG. A KG comprehends user interest by understanding the semantics of entities and relations it has ((Cao et al., 2019); (Ma et al., 2019); (Wang H. et al., 2018); (Wang H. et al., 2018a); (Zhang F. et al., 2016); (Zhang Y. et al., 2018)).

After LFMs, the next effort in the recommendation engine is the relevancy of the KG. A KG represents large-scale information from multiple domains (Zou, 2020). A KG represents nodes as entities and edges as relations between entities and follows the Resource Description Framework (RDF) standard (Purohit et al., 2021). Edge

represents a triple (head entity, relation, tail entity), i.e., a fact in the graph, implying the specific relationship between the head entity and tail entity. For example, ("Joe Biden", "president of", "America") indicates that Joe Biden is the president of America. A KG is a heterogeneous network containing multiple categories of nodes and relations. Such a graph has a strong representation ability since many attributes of an entity can be obtained by following different edges. These relational links help to identify the high-level relations of entities. In 2012, Google introduced the KG into the search framework for a better understanding of the query and to make search results more user-friendly (Fensel et al., 2020). KGs have been created and applied in multiple scenarios, including search engines, RSs, Question Answering systems, etc (Guo et al., 2020). The KG improves prediction accuracy and enhances the explainability capability of the models.

2.5.9 Advantages and Limitations

KGs offer structured representation and contextual understanding by capturing relationships between entities, enabling sophisticated querying and semantic analysis. They enhance data integration and inference, supporting better decision-making while aiding entity resolution and data exploration. This helps to improve the precision of recommendations. Another benefit of the KG-based RS is the explainability of recommendation results. However, they require significant effort for data acquisition, integration, and expertise in modelling and face challenges of scalability, subjectivity, and privacy concerns, which organizations should carefully consider based on their specific needs and resources (Wang H. et al., 2018).

2.5.10 Summary of the Section

The preceding section has conducted an exhaustive examination of the KG concept, encompassing a meticulous definition and a profound comprehension of KGs, enhanced by vivid illustrative examples. Furthermore, this study has delved into relevant concepts within this realm, including HINs, KGs, meta-paths, meta-graphs, H-hop neighbours, and the incorporation of user feedback.

The following section will explore the domain of KGE and its associated principles.

2.6 KNOWLEDGE GRAPH EMBEDDING (KGE)

A KG $\mathcal{G}\text{know} = (V, E)$ constitutes entities and relationships as two primary graph components. Due to the nature of HIN, the KG may contain enormous amounts of data and face performance issues. While KGs prove proficient in representing structured factual data, they are difficult to manipulate due to the large-scale and complicated graph structure. Consequently, the challenge lies in effectively and efficiently extracting and harnessing valuable information from extensive KGs. This becomes particularly crucial for subsequent tasks such as link prediction (Sun et al., 2019), entity classification (Yu et al., 2022), and recommendation generation (Xian et al., 2019).

To address this formidable challenge, the ML community has introduced the KGE technique, which has garnered considerable attention (Sun et al., 2019). The core concept behind KGE is to embed entities and relationships within a KG into a lower-dimensional space, typically represented as vectorial embeddings. These embeddings preserve the semantic meaning and relational structure of the original KG. Subsequently, the learned embeddings for entities and relationships are leveraged to tackle downstream tasks, including KG completion (Bordes et al., 2013), question answering (Zheng et al., 2021), and entity classification (Yu et al., 2022).

Overall, KGE embeds a KG into $\mathcal{d}$-dimensional space. The word embedding means vectorisation of the information into low $\mathcal{d}$-dimensional vectors. The low-dimensional embedding still preserves the inherent property of the graph and can be quantified by semantic meaning or high-order proximity (Cai et al., 2018).

This study comprehensively explores the KGE methodologies categorized under embedding-based, path-based, and unified methods in the subsequent subsections.

2.6.1 Implementation Methodologies

The learning paradigm for KGEs comprises three essential components: embedding mapping, score function, and representation training (Cao et al., 2022).

- The embedding mapping projects entities and relations into lower-dimensional vectors, representing them in a mathematical space. Various mathematical spaces have been employed for this purpose.

- The score function, denoted as $s(\cdot)$, operates on the mapped embedding space and gauges the credibility of a triple. It is defined on the embedding vectors of a triple, s(h, r, t), aiming to assign higher scores to true triples and lower scores to false ones. Different score functions arise from distinct representation spaces.

- The representation training phase aims to learn entity and relation embeddings by maximizing scores for positive triples and minimizing scores for negatives. As acquiring precise positive and negative triples is impractical, observed triples are commonly considered in the KGEs field.

2.6.2 Embedding-based Methods

Embedding-based approaches commonly utilize KG information directly to enhance the representation of entities or users. To effectively harness KG data, the use of KGE algorithms becomes imperative, as they encode the KG into low-dimensional embeddings. KGE algorithms fall into two primary categories (Wang et al., 2017):

- **Translation Distance Models:** This category includes algorithms like TransE (Bordes et al., 2013), TransH (Wang Z. et al., 2014), TransR (Lin et al., 2015), TransD (Ji et al., 2015), and similar methods. These models are centered around the idea of representing relationships as translations between entity embeddings in the embedding space.

- **Semantic Matching Models**: Another class involves semantic matching models such as DistMult (Yang et al., 2015), which focuses on capturing the relationships between entities through a bilinear product in the embedding space.

These categorizations provide a foundational framework for understanding the diverse landscape of KGE algorithms, each contributing to the broader objective of effectively encoding KG information into compact embeddings.

2.6.3 Embedding-based Methods – Advantages and Limitations

Entity embedding serves as the cornerstone of embedding-based methods in RSs. These methods involve transforming entities within a KG into continuous vector representations or embeddings. These embeddings encode semantic information about the entities and their relationships, enabling RSs to comprehend the latent features of items and users. By learning these embeddings, the recommendation framework can operate effectively in a continuous space, facilitating tasks such as similarity calculations and personalized recommendations. Despite their effectiveness in capturing semantic meanings, embedding-based methods often fall short in capturing the intricate connectivity patterns of the graph and explaining recommendation rationale. This has spurred a drive to combine them with path-based techniques, creating a more comprehensive approach encompassing both the advantages of embeddings and the explanatory power of graph patterns (Sun et al., 2020).

2.6.4 Connectivity Path-based Methods

Path-based methods for RSs leverage the connectivity patterns in a user-item graph. These methods are often referred to as recommendations in a HIN. Path-based recommendation methods in HINs take advantage of these multiple types of entities and relationships to enhance recommendation accuracy (Sun et al., 2011) (Guo et al., 2020). Here is how they typically work:

- **Path Generation**: Path-based methods generate paths between user and item nodes in the graph. These paths are sequences of nodes and edges that connect

users and items through a sequence of intermediate nodes of diverse types. Paths can capture complex relationships between users and items beyond direct interactions.

- **Path Similarity**: The similarity between users or items is calculated based on the similarity of the paths connecting them in the graph. Paths that share similar patterns of entities and relationships indicate potential similarity in user preferences or item attributes (Sun et al., 2011).

- **Recommendation Generation**: Once the path similarities are computed, recommendations are generated by identifying items that are similar to the ones the user has interacted with or shown interest in. This can be done by considering items connected to similar users or paths similar to the ones the user has taken.

Path-based methods are particularly useful in scenarios involving multiple types of entities and relationships, allowing them to capture nuanced connections and recommendations. However, they can also be challenging to implement and tune effectively due to their complexity.

2.6.5 Path-based Methods – Advantages and Limitations

Path-based KGE techniques provide a means to capture intricate relationships and multi-hop semantics within KGs, offering advantages such as expressive representation of complex connections, handling of missing data, and contextual understanding. However, their computational complexity, susceptibility to noise in semantic interpretation, and challenges in determining optimal path lengths can limit their applicability. Balancing interpretability and complexity while addressing scalability and data sparsity issues remains crucial for their effective utilization in learning from structured knowledge (Troussas & Krouska, 2022).

2.6.6 Unified Methods

Embedding-based methods and path-based methods are two distinct approaches that utilize different aspects of information in a KG for RSs. Embedding-based methods focus on capturing the semantic representation of users and items in the KG, while path-based methods emphasize the use of semantic connectivity information. However, for more effective recommendations, unified methods have been developed to integrate both aspects of information. These unified methods, often employing embedding propagation, aim to enhance entity representation by leveraging the connective structure within the KG. By combining semantic representation and connectivity information, these approaches seek to leverage the full potential of KG data for improved recommendation accuracy and utility (Guo et al., 2020).

2.6.7 Unified Methods – Advantages and Limitations

Unified methods that integrate both embedding-based and path-based approaches have the potential to provide more accurate and contextually rich recommendations. However, they come with increased complexity, potential resource requirements, and challenges related to data quality and interpretability. The effectiveness of such methods would depend on the specific characteristics of the KG, the recommendation task, and the available computational resources (Guo et al., 2020).

2.6.8 Summary of the Section

In the realm of KG-based RSs, two main approaches have emerged. The first, known as embedding-based methods, involves converting entities and relationships within a KG into continuous embeddings, which are then integrated into recommendation frameworks. However, these methods tend to overlook the valuable connectivity patterns in the graph, leading to a lack of detailed explanations for recommendations. On the other hand, path-based methods leverage the user-item graph to identify similarities through connective patterns, providing not only more transparent and XR but also insight into the underlying reasons for those suggestions. A current research trend involves unifying these two approaches, allowing for the comprehensive

utilization of both semantic embeddings and informative graph patterns, resulting in more accurate recommendations with meaningful explanations for users.

The next section will explore the domain of RL and its associated principles.

2.7 REINFORCEMENT LEARNING

In the realm of RL, an independent agent enhances its performance in a designated task by engaging with its environment. According to (Russell, 2010), an agent can be defined as a system that perceives its surroundings through sensors and enacts changes in the environment through actuators. Unlike receiving explicit instructions from an expert, RL agents gauge their performance based on a reward function denoted as R. As the agent navigates through various states, it selects actions and occasionally receives rewards from the environment as a consequence of its decisions. The primary objective for the agent is to maximize the cumulative rewards it garners over its operational duration. This is achieved by progressively refining its understanding of the expected utility, which involves the discounted sum of anticipated future rewards across different state-action pairs. A central challenge in this process is finding the right balance between exploration and exploitation. The agent is compelled to exploit its current knowledge by picking actions that are proven to yield high rewards. However, it must also venture into uncharted territory to unearth actions with potentially higher rewards than those currently deemed optimal for each given state. In essence, the learning agent must strike a harmony between capitalizing on its existing knowledge to secure rewards and delving into the unknown to make more informed action choices in the future (Kiran et al., 2021).

In the subsequent subsections, this study reviews a comprehensive exploration of the RL methodologies.

2.7.1 Markov Decision Process (MDP)

MDPs are considered the de-facto standard when formalizing sequential decision-making problems involving a single RL agent, in which actions affect the current short-term rewards, the subsequent states, and future rewards (Yan et al., 2020). An

MDP represents a tuple $<S, A, T, Rw, \gamma>$, where S is the set of all possible states, which is the generalization of the environment, A is the set of all possible actions of the agent that can be adopted in the states, T is a transition function, Rw is a reward function, and γ is a discount factor (R. Wickman, 2022).

Here, the reward function R explains the rewards that the environment returns to the agent after executing an action.

$$R: S \times A \times S \rightarrow R \quad \ldots(2.11)$$

Similarly, the state transition probability function T can be explained as

$$T: S \times A \times S \rightarrow p(S) \in (0, 1) \quad \ldots(2.12)$$

Here, $\gamma \in [0, 1]$ denotes the discount factor, which can be treated as a hyperparameter of the agent and serves to promote the faster availability of short-term rewards to the agent.

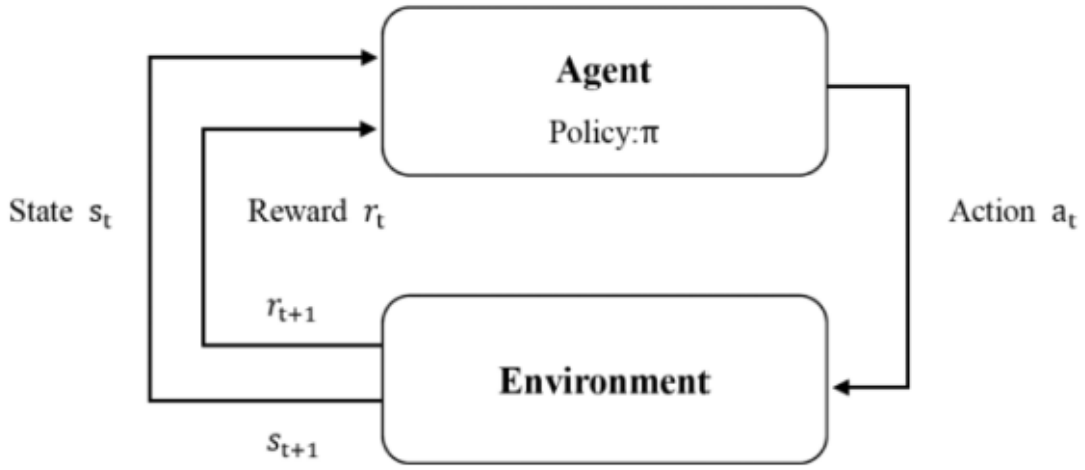

Figure 2.15 The Interaction Process between Agent and Environment

When in any state, $s \in S$, selecting an action $a \in A$ will result in the environment entering a new state $s' \in S$ with a transition probability $T(s, a, s') \in (0, 1)$, and give a reward $R(s, a, s')$. This process is illustrated in **Figure 2.15**.

The goal is to find a policy π that maximizes the expected return/action-value function $Q(s, a)$, the target policy is defined as (2.13) (Plaat et al., 2023).

$$\pi^* = \underset{\pi}{argmax}\ Q(s, a)$$

$$\pi^* = \underset{\pi}{argmax}\ \mathrm{E}_{\pi,\mathbb{T}}\left[\sum_{k=0}^{K} \gamma^k\ r_{t+k}\ |\ s_t = s, a_t = a\right] \quad \ldots(2.13)$$

where π^* is better or equal to all other policies.

2.7.2 Reinforcement Learning Framework

RL is another big area, and many successful applications belong to this area, e.g., Alphago (Silver et al., 2016), self-driving cars (Chopra & Roy, 2020), etc. The RL framework comprises Environment State, Agent, Actions, and Rewards. The agent takes an appropriate action in a state to maximize the rewards and go to another state. This framework belongs to an environment and demonstrates the ability of the agent to understand it. The agent understands high-level causal relationships.

There are many methods to implement the RL. In the following sections, this study will briefly review the same.

2.7.3 Q-Learning

Q-learning (Watkins & Dayan, 1992), an influential outcome of early RL research, is an off-policy and Temporal Difference (TD) learning algorithm. It operates by directly updating values within a Q-table to attain the optimal policy, using a target policy for this purpose. Meanwhile, a separate behaviour policy, often employing the

ε-greedy strategy, facilitates semi-random exploration of the environment. The primary objective of Q-learning is to approximate the optimal action-value function, q∗, through direct estimation. This function signifies the highest expected cumulative reward achievable for actions in specific states.

The Q-learning algorithm can be formulated as follows.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha * \left[R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \quad \ldots(2.14)$$

where the equation denotes that the state s_t explores the environment with a behaviour policy based on the values in the Q-table at timestep t, i.e., it performs action a, the reward R and a new state s_{t+1} is obtained based on the feedback from the environment. The equation is executed to update to obtain the latest Q-table, it will continue to update the new state s_{t+1} after completing the above operation until the termination time t.

2.7.4 REINFORCE

REINFORCE (Williams, 1992) takes a distinct approach by eschewing direct optimization within the policy space. Instead, it focuses on learning a parameterized policy without the intermediary step of estimating values. This algorithm uses the Monte Carlo method to learn policy parameters based on estimated returns and full traces. The technique involves constructing a policy using neural networks. These networks accept states as inputs and produce probability distributions in the operational space as outputs.

The policy π is parameterized with a set of weights θ so that $\pi(s;\ \theta) \equiv \pi(s)$, which is the action probability distribution on the state, and REINFORCE is updated with:

$$\triangle \omega_{i,j} = \alpha_{i,j} \left(r - b_{i,j} \right) \frac{\vartheta}{\vartheta \omega_{i,j}} \ln g_i \quad \ldots(2.15)$$

where $a_{i,j}$ is a non-negative learning factor, r denotes the discounted reward value, and $b_{i,j}$ is a representation function of the state for reducing the variance of the gradient estimate. (Williams, 1992) points that $b_{i,j}$ could have a profound effect on the convergence speed of the algorithm. g_i is the probability density function for randomly generating unit activation-based actions.

2.7.5 Actor-Critic

The Actor-Critic algorithm (Konda & Tsitsiklis, 1999) is an RL approach that leverages a parameterized policy (Actor) and a value function (Critic) to enhance learning efficiency and stability. By estimating the expected cumulative reward for a given state-action pair, the value function guides the optimization of the policy through policy gradient methods, enabling the Actor to make more informed decisions. This amalgamation of policy-based and value-based techniques allows the algorithm to strike a balance between effectively exploring the environment to maximize rewards and accurately assessing the desirability of different states and actions.

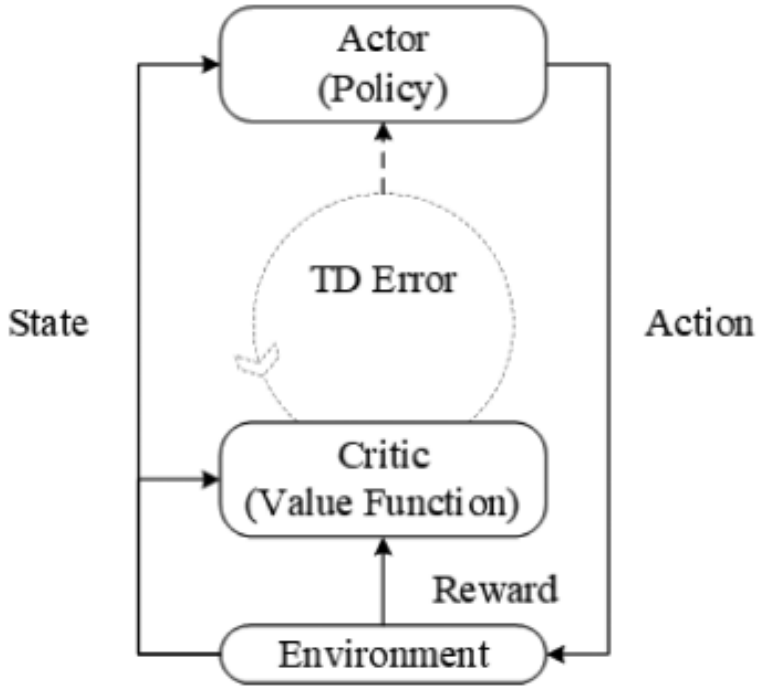

Figure 2.16 The Framework of Actor-Critic Algorithm

Figure 2.16 is the framework of the Actor-Critic algorithm. Actor receives a state from the environment and selects an action to perform. Meanwhile, Critic receives the current state and the state generated by the previous interaction and calculates the TD error to update Critic and Actor.

The Actor-Critic architecture is widely employed as the basic framework for RL algorithms.

2.7.6 Reward Shaping

The reward function is an essential function in the RL framework and leads to a better recommendation score. The design of the reward function is crucial. The higher reward function value leads to an optimal path for that recommended product. The KGE assists in evaluating the user's recommendation score for the product in the KG environment.

RL agents are driven to maximize their cumulative reward, and the best policy in a given context is determined based on the underlying reward function. However, practical scenarios often involve challenges like limited and delayed rewards. RL agents primarily rely on this reward signal to navigate their environment, which can lead to slow learning. To address this, supplementary information can be introduced through a shaping reward and added to the intrinsic environment rewards. This shaping reward aims to expedite learning and enhance final performance by providing additional guidance to the agent. This concept is known as reward shaping (Kiran et al., 2021).

2.7.7 RS Implementation with Advances in RL

RSs use RL techniques in a few use cases, such as news recommendations (Song et al., 2021) and post-hoc explainable (Wang et al., 2022). Researchers have also explored RL in KG settings for tasks such as question answering (QA) (Das et al., 2017); (Cui et al., 2023); (Xiong et al., 2017)), which formulates multi-hop reasoning as a sequential decision-making problem. (Xian et al., 2019) propose RL-based explicit reasoning over KGs for recommendation with interpretation, but scalability

over large real-world KGs seems to be an issue. (Song et al., 2019) proposes a mechanism to generate meaningful paths from users to relevant items by learning a walking policy on the user-item-entity graph and producing much-enhanced output. It has an issue with proper reward assignments and scalability.

2.7.8 Summary of the Section

RL is a machine learning paradigm where an agent learns to make a series of decisions through interactions with an environment, aiming to maximize its cumulative rewards. The agent's objective is to acquire a policy that guides its actions based on states to achieve the highest long-term rewards, striking a balance between exploring new actions and exploiting known high-reward actions. Key components include states, actions, rewards, value functions, and Q-functions, all integrated within the framework of Markov Decision Processes. Q-learning, REINFORCE, and Actor-Critic are primary RL methods, and reward shaping plays a significant role. These methods, often combined with deep neural networks, enable RL to address intricate, high-dimensional state spaces and find applications across diverse domains.

The following section will explore the XR and its associated principles.

2.8 EXPLAINABLE RECOMMENDATIONS

Personalized recommendations have become integral to our online experiences in the modern digital landscape. Whether suggesting movies on streaming platforms, products on e-commerce websites, or content on social media, RSs play a pivotal role in guiding users through a sea of options. However, as these systems have grown in complexity, concerns have arisen regarding their lack of transparency and the potential for algorithmic bias. This has led to the emergence of "explainable recommendation" approaches, which aim to provide accurate suggestions and clear and understandable explanations for those recommendations.

XRS address the black-box nature of traditional recommendation algorithms, which often operate behind the scenes, leaving users in the dark about how suggestions are generated. This lack of transparency can lead to user distrust,

frustration, and even ethical issues when biases in the data propagate through the recommendations. XR seeks to bridge this gap by providing users with insights into why a particular item or content was recommended to them, fostering a sense of trust, and enhancing the overall user experience.

In this realm, the challenge lies in striking a balance between accuracy and comprehensibility. On one hand, recommendations must remain accurate and relevant to users' preferences, while on the other, the explanations must be presented in a format that users can easily grasp. Researchers and developers in the field are exploring various techniques, such as model visualization, feature importance analysis, and natural language explanations, to make the recommendation process more transparent and interpretable.

XRS are not only beneficial for users; they also have implications for businesses and organizations. Providing explanations for recommendations can help companies build stronger relationships with customers, as users feel more in control of their choices. Moreover, these systems can aid in complying with regulations and standards related to data privacy and fairness, which have become increasingly important in today's data-driven world.

This evolving landscape represents a convergence of fields such as machine learning, human-computer interaction, and ethics. As researchers delve deeper into designing RSs that are not only accurate but also understandable, the future of personalized content delivery holds the promise of greater transparency, user satisfaction, and ethical responsibility.

The XR was formally introduced by (Zhang et al., 2014a). The main problem with complex models like LFMs is the explainability of the model recommendations. It necessitates the need for XR ((Zhang & Chen, 2020) (Zhang et al., 2014a)) efforts in further recommendation research.

The remaining subsections will delve into XR's specific challenges and issues. These challenges will be addressed by harnessing the advancements in KG, KGE, and RL

techniques to effectively achieve the overarching objectives and goals of creating transparent and comprehensible RSs.

2.8.1 KG-based Approach for XRS

Compared with traditional RSs, KG-based recommendations make the reasoning process available. This section will leverage KGs and their variants for an XR. To achieve the objective of XAI, researchers applied KGs and integrated them into RSs as side information (Guo et al., 2020). They exploited the information in the KG to improve the recommendation performance and provided recommendation explainability due to the intuitive ease of understanding relationships between entities (Wang H. et al., 2018).

The prominent KG-based implementation methods for RSs are KG embedding-based (Zhang F. et al., 2016), Post-hoc implementation (Perdih et al., 2021), Path-based (Xian et al., 2019), and Unified approaches (Wang H. et al., 2018) as illustrated in **Figure 2.17**. The example shows a small snippet of **Figure 2.14**. It demonstrates the functioning of different kinds of embeddings in the KGE environment, where entities such as users, products, and others, as well as related relations, are embedded into low-rank embeddings.

In KG embedding-based methods, the logic of explainability embeds into the recommendation generation process (Zhang F. et al., 2016). That line of research focuses on making recommendations using KGs embedding models, such as entity2rec (Palumbo et al., 2017) and node2vec (Grover & Leskovec, 2016). The post-hoc explanation assists in generating the recommendation reasoning through the soft matching algorithms after the model development. (Ai et al., 2018) proposed to enhance the CF method over KG embedding for personalised recommendation, followed by a soft matching algorithm to find explanation paths between users and items.

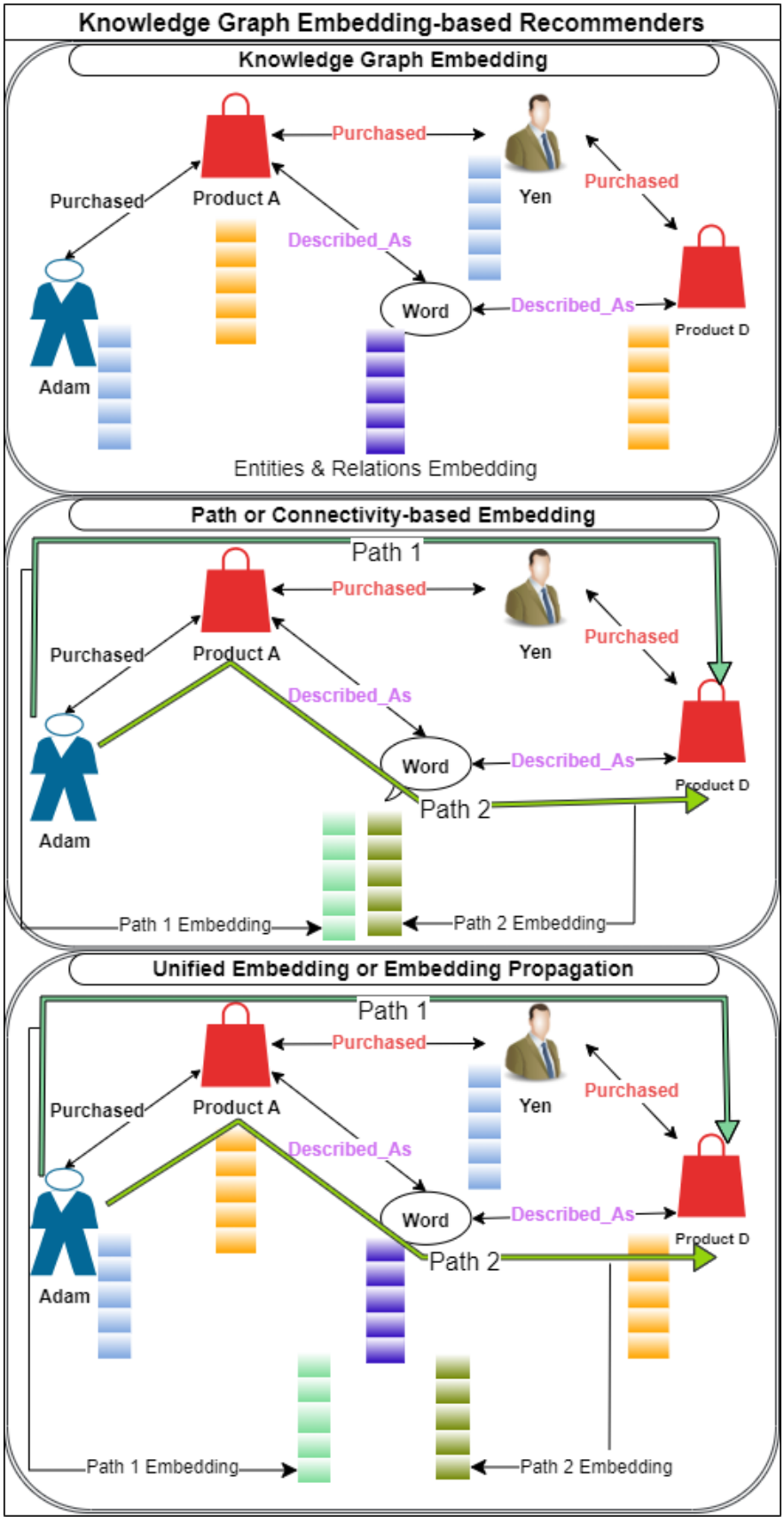

Figure 2.17 **Knowledge Graph-based Recommender Systems**

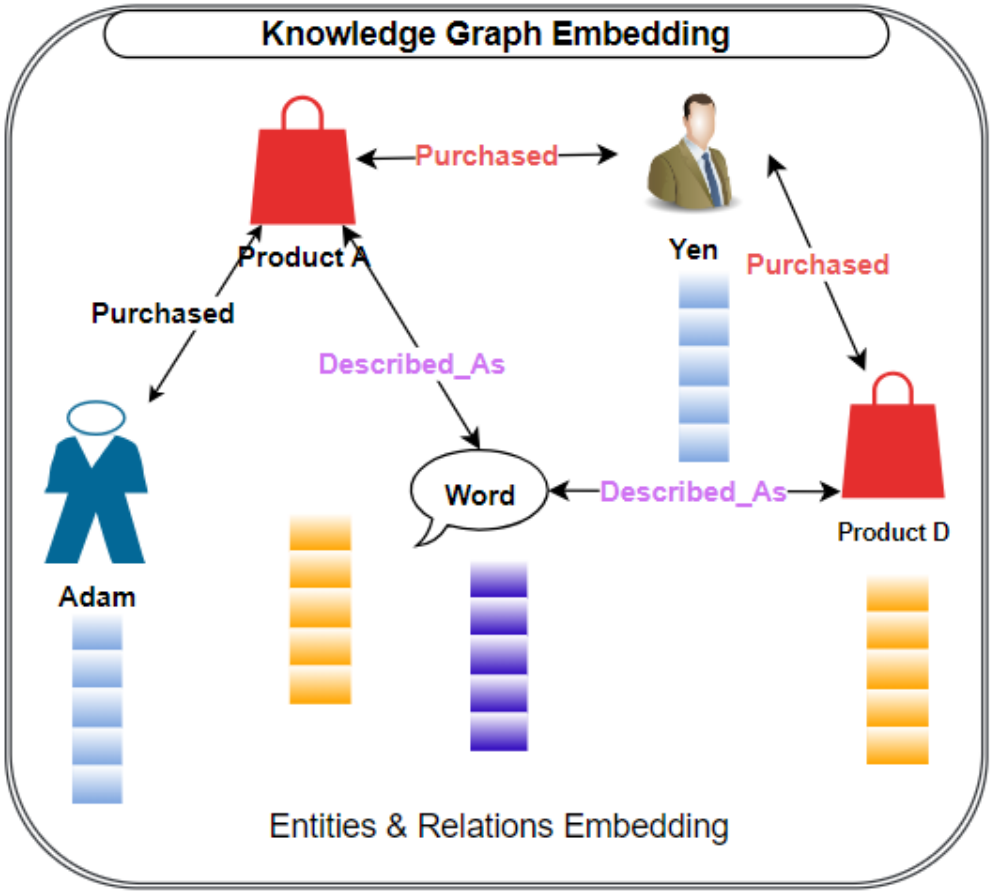

Figure 2.18 Knowledge Graph Embedding-based Recommender Systems

Path-based recommendations follow the KG and its meta-paths. (Gao et al., 2018) proposed the notion of meta-paths to reason over KGs. (Xian et al., 2019) proposed a Policy-Guided Path Reasoning (PGPR) algorithm to use RL to search for reasonable paths between user-item pairs.

The unified approach focuses on unifying the embedding-based methods with the path-based methods to exploit information from both sides (Guo et al., 2020). These methods benefit from both the semantic embedding and the path patterns of the KG.

a. KGE-based Methods Used for XRS

Figure 2.18 illustrates the embedding-based methods, which generally use the information from the KG directly to enrich the representation of items or users. The example shows a small snippet of Figure 2.14. It demonstrates the functioning of

embeddings in the KGE environment, where entities such as users, products, and others, as well as related relations, are embedded into low-rank embeddings. KGE models find the similarity between entities by calculating their representation distance (Zhang F. et al., 2016).

An item graph is a kind of KG that forms graphs with items and their related attributes. KGE algorithms leverage this item graph to encode the graph for a more comprehensive item representation and integrate the item side information into the recommendation framework. The user-item graph forms graphs with users, items, and related attributes. Attribute-related relations (brand, category, etc.) and user-related relations (co-buy, co-view, etc.) serve as edges.

Most embedding-based methods (Huang et al., 2018) build KGs with multiple item side information to enrich the representation of items that form item graphs. This item graph-based KG information assists in modelling the user representation more precisely. Some models ((Zhang Y. et al., 2018); (Ai et al., 2018)) build user preferences through user-item KGs by introducing users to the graph design. Entity embedding is the core of embedding-based methods, and some research refines the embedding with Generative Adversarial Networks (GAN) (Yang et al., 2018) or BEM (Ye et al., 2019) for a better recommendation. Embedding-based methods intrinsically leverage the information in the graph structure. These embedding methods lack in identifying multi-hop relational paths. (Ai et al., 2018) formulates a CF approach over KG embedding and later adopts a soft matching algorithm to find explanation paths between users and items. It is a post-hoc explanation. The problem with this approach is the explanation generation process.

b. Connectivity Path-based Methods Used for XRS

Figure 2.19 illustrates the Connectivity path-based methods, also known as HIN-based recommendations, leverage semantic similarities of entities in different meta-paths as the graph regularization to refine the representation of users and items in the HIN. The example exhibits a small snippet of **Figure 2.14** and demonstrates the functioning of path-based embeddings, where the traversals from the user node to the product nodes are embedded as low-rank embeddings. Connectivity embedding is a

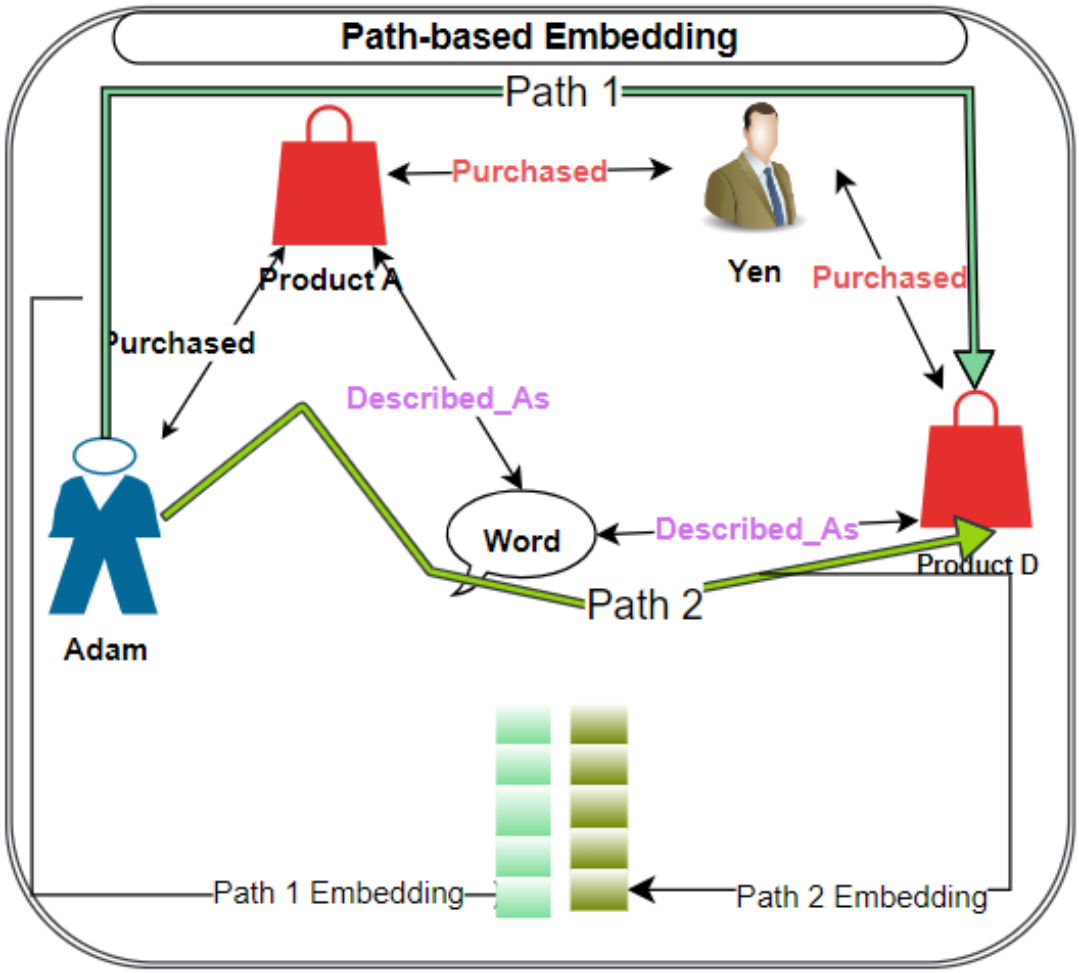

Figure 2.19 Connectivity Path-based Recommender Systems

core component of path-based methods. (Yu et al., 2013) proposed the Hete-MF, which extracts L different meta-paths and calculates item-item similarity in each path. Later, (Luo et al., 2014) proposed Hete-CF to find the user affinity to unrated items by taking the user-user, item-item, and user-item similarity together as regularization terms. Therefore, the Hete-CF outperforms the Hete-MF model. Later, (Yu et al., 2014) proposed HeteRec-p, which further considers the importance of different meta-paths for different users. "HeteRec-p" first clusters users based on their past behaviours into c groups and generate personalized recommendation with the clustering information instead of applying a global preference model. To overcome the limitation of the meta-path representation ability, (Zhao et al., 2017) designed FMG by replacing the meta-path with the meta-graph. Since a meta-graph contains richer connectivity information than a meta-path, FMG captures the similarity between entities more accurately.

Recently, some frameworks have been proposed to learn the explicit embedding of paths that connect user-item pairs to model the user-item relations. For instance, (Hu et al., 2018) proposed MCRec, which learns the explicit representations of meta-paths to depict the interaction context of user-item pairs. (Sun et al., 2018) proposed a recurrent knowledge graph embedding (RKGE) to define the meta-paths automatically. (Wang et al., 2019) proposed a knowledge-aware path recurrent network (KPRN) solution. KPRN constructs the extracted path sequence with the entity and the relation embeddings and encodes it with an LSTM layer. A sequential RNN model has been developed over the qualified paths between user-item pairs to predict the pair ranking score. This approach assisted in improving the recommendation performance but has limitations around scalability. It is not feasible to explore entire paths in large real-world KGs.

(Xian et al., 2019) proposed Policy-Guided Path Reasoning (PGPR) to use RL to search for reasonable paths between user-item pairs. They formulated the recommendation problem as a Markov decision process. They trained an agent on sample paths between users and items by carefully designing the path-searching algorithm, the transition strategy, terminal conditions, and RL rewards. PGPR predicts the recommendations for users with specific paths to interpret the reasoning process. Later, (Song et al., 2019) proposed a similar model (EKar*) by adopting the RL technique in generating recommendations.

Path-based methods generate recommendations based on user-item graphs, also known as HIN-based recommendations. Traditional path-based methods ((Zhao et al., 2019); (Shi et al., 2019)) generally integrate MF with extracted meta-paths in the HINs. These methods utilize path connectivity to regularize or enrich the user-item representation. The disadvantage of these methods is that they commonly need domain knowledge to define the type and number of meta-paths. With the development of DL techniques, different models ((Xian et al., 2019); (Huang et al., 2019)) proposed the encoding of the path embedding explicitly.

Path-based methods offer inherent interpretability by matching similarity on the meta-paths. In general, these models take advantage of the connectivity similarity

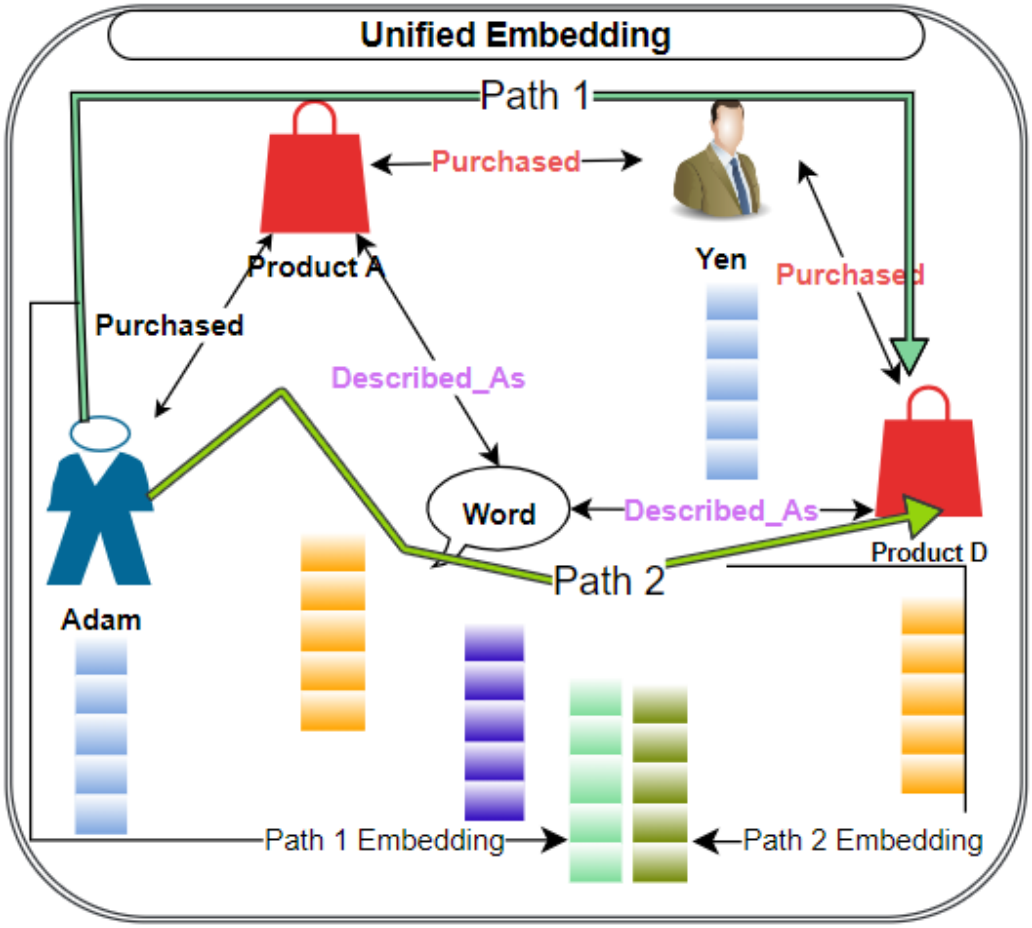

Figure 2.20 Unified Embedding Recommender Systems

of users and items to enhance the recommendations. However, these approaches may face scalability challenges in a large real-world KG.

c. Unified or Embedding Propagation Methods Used for XRS

Figure 2.20 illustrates the Unified methods, which leverage the semantic representation of entities and relations along with their connectivity information. The example exhibits a small snippet of Figure 2.14 and demonstrates the functioning of embedding propagation to refine entity representations by considering multi-hop neighbours. These methods refine the entity representation with the guidance of the connective structure in the KG.

Unified methods benefit from the interpretability of path-based methods and use the propagation process to discover user preference patterns. (Wang H. et al.,

2018) proposed RippleNet, which is the first work to introduce the concept of preference propagation. Specifically, RippleNet first assigns entities in the KG with initial embeddings. Then, it samples ripple sets from the KG. (Tang et al., 2019) proposed AKUPM, which models users based on their click history. AKUPM first applies TransR to entity representation. During the propagation process, AKUPM learns the relations between entities with a self-attention layer and propagates the user preference toward different entities with bias. Finally, embeddings from different-order neighbours of interacted items aggregate with the self-attention mechanism to obtain the final user representation. Later, (Li et al., 2019) extended the AKUPM and designed RCoLM. RCoLM trains the KG completion and the recommendation module jointly. It assumes an item should have a similar latent representation in the two modules. With that assumption, RCoLm unifies two modules and facilitates their mutual enhancement. Thus, RCoLM outperforms the AKUPM model.

(Wang et al., 2019) proposed KGCN. They model the final representation of a candidate item by aggregating the embedding of entities in the KG from distant neighbours of the candidate item. RippleNet and KGCN are two similar frameworks. RippleNet is used to propagate the user preference from historical interests outwardly, while KGCN learns item representations from distant neighbours inwardly. Moreover, KGCN leverages the idea of GCN by sampling a fixed number of neighbours as the receptive field, which makes the learning process highly efficient and scalable.

Recently, some research has explored the propagation mechanism in the user-item graph. (Wang et al., 2019) proposed KGAT. They model the high-order relations between users and items with embedding propagation. KGAT first applies TransR to obtain the initial representation for entities. Then, it runs the entity propagation from the entity itself outwardly. During the outward propagation process, entity information will interact with the multi-hop neighbours iteratively.

(Zhao et al., 2019) proposed IntentGC. They exploit rich user-related behaviours in the graph for a better recommendation. They designed a faster graph-convolutional network (GCN) to guarantee the scalability of IntentGC. Recently, (Sha et al., 2021) proposed AKGE, which learns the representation of user and candidate

items by propagating information in a subgraph of this user-item pair. AKGE first pre-trains the embeddings of entities in the graph with TransR, then samples paths based on the pairwise distance. Next, AKGE uses an attention-based Graph Neural Network GNN to propagate the information from neighbours for the final representation of this user-item pair. The construction of the subgraph removes fewer related entities in the graph, facilitating the mining of high-order user-item relations for the recommendation.

Unified methods benefit from both the semantic embedding of the KG and semantic path patterns. These methods leverage the idea of embedding propagation to refine the representation of the item or user with multi-hop neighbours in the KG. These works generally adopt a GNN-based architecture that naturally fits the process of embedding propagation. Such methods have been a new research trend. Unified methods inherit interpretability from path-based methods. The propagation process discovers the user preference patterns in the KG, similar to finding connectivity patterns in path-based methods.

2.8.2 RL-based Methods Used for XRS.

RL has gained attention in recent research for its applicability to multi-hop reasoning and path-finding tasks. Researchers such as (Xian et al., 2019), (Song et al., 2019), (Xu et al., 2021), and (Wang et al., 2022) have introduced RL-based methods like Policy-Guided Path Reasoning (PGPR), Explainable knowledge aware recommendation (EKAR), Knowledge Graph Deep Q-Network (KGDQN), and Reinforcement learning framework for Multi-level recommendation Reasoning (ReMR) respectively, to formulate recommendation problems as Markov decision processes. By training agents with path-searching algorithms, transition strategies, terminal conditions, and RL rewards, these approaches enable the generation of transparent paths connecting users and recommended items. The benefit of adopting RL is the discovery of reasonable traversals that connect the user with the recommended items, making the system more transparent.

In their research, (Xian et al., 2019) present PGPR, a novel approach that leverages RL to autonomously identify effective multi-hop paths within a KG between

user-item pairs, aiming to enhance RSs. This is achieved by framing the recommendation task as an MDP, in which an RL agent is trained to navigate the KG using a customized path-searching algorithm, transition strategy, terminal conditions, and reward framework designed to reinforce accurate connections between user-item pairs. An intriguing aspect of the approach lies in its transparency, attributed to the direct availability of the connecting paths, contributing to improved interpretability. Nonetheless, the method encounters several challenges, such as high computational demands arising from intricate KG structures, potential limitations in tailoring recommendations to individual preferences, the possibility of suboptimal suggestions, and a potential oversight of relevant factors beyond the sole consideration of path probabilities in the reasoning process. These challenges underscore the intricacies of employing RL for path discovery in KGs and prompt the need for further investigation to address these limitations and refine the approach's overall effectiveness in real-world recommendation scenarios.

In their subsequent investigation, detailed in the research (Song et al., 2019), the authors introduced EKAR, a novel methodology that leverages the power of deep RL within an MDP framework to construct intelligible pathways guided by user preferences. While EKAR demonstrates promise, its implementation requires a substantial volume of personalized policy training data to achieve optimal performance. Notably, the approach suffers from certain limitations, such as the absence of an intermediary reward mechanism, wherein binary rewards (+1, 0, or -1) are assigned based on the fulfilment status of paths involving terminal entities. Consequently, EKAR faces notable challenges in delivering refined product recommendations and their accompanying explanations, as its decision-making process heavily relies on path probabilities or rewards to determine the prioritization of products for recommendation.

Xu's work introduces KGDQN, a method that integrates KGs into a deep Q-network (DQN) framework for item recommendations. The DQN uses an RL algorithm to solve tasks by learning optimal actions through trial and error. However, the approach inherits drawbacks from the DQN model, including issues of overestimation bias and training instability. Although Xu proposes a non-binary

rewards strategy, which means that the rewards given in this approach are not just simple binary values (like 0 or 1) but can take on different values to provide more nuanced feedback, the method still faces challenges in determining optimal recommended products and the associated reasoning paths. These paths are established by seeking to maximize the multiplication of rewards with corresponding probabilities. These challenges underscore the complexity of effectively leveraging KGs within a DQN framework for item recommendations, necessitating further research to overcome limitations and enhance the method's practical utility.

In a recent development, (Wang et al., 2022) introduced a method known as ReMR, which capitalizes on ontologies and instance KGs to effectively represent user interests spanning multiple levels, thereby capturing intricate hierarchical relationships. By integrating multi-level reasoning paths, ReMR significantly enriches the modelling of user preferences, culminating in the generation of enhanced recommendations. This approach showcases the potential of leveraging KGs to provide more sophisticated and refined RSs through the incorporation of hierarchical information.

2.8.3 Datasets for KG-based RSs

The most used datasets for the XRS are as follows:

a. Movie:

- **MovieLens** -The most used datasets are the MovieLens benchmark datasets (GroupLens, Movielens dataset, 1997), collected from the MovieLens website (MovieLens, 1997), including MovieLens-100K, MovieLens-1M, and MovieLens-20M with a different number of ratings. Each dataset contains ratings, the movie’s attributes, and the user’s profile. MovieLens-100K is the smallest one, which was evaluated by some early works and works with high computational complexity. MovieLens-1M is the most popular one with a balanced rating number and density, while MovieLens-20M is the largest one, which is suitable for verifying the scalability of the model. Note that the movie recommendation datasets are much denser than other scenarios.

- **DoubanMovie** - Besides the MovieLens dataset, there is also the DoubanMovie dataset (Yang et al., 2018) crawled from Douban (Douban, 2005), a popular Chinese social media network. DoubanMovie maintains the movie tag and can link the movie title with the entity in the Chinese KG, CN-DBPedia, to enrich the representation of items.

b. Book:

- **Book-Crossing** - Book-Crossing (Ziegler et al., 2005) contains the user's demographic information and the book's attributes, such as the author, the publisher, and the year of publication.

- **Amazon-Book** - The Amazon-Book dataset (McAuley et al., 2015) is the largest subset of the Amazon Review dataset. Compared with the Book-Crossing dataset, the user's review, and user's behavior relations, such as "also viewed," "viewed and bought," "also bought," and "bought together" are available. Therefore, more relations can be considered in the constructed KG. Moreover, the Amazon-Book dataset is much larger.

c. Music:

- **Last.fm** - There are two datasets extracted from the Last.fm online music system (Last.fm, 2002). Last.FM-a (GroupLens, Last.fm-A dataset, 2011) is a small dataset, which provides the user's social relation, tags of tracks and artists, and listening records. While Last.FM-b (Schedl, 2016) contains demographic information of users, tags of tracks, and user's listening records.

- **KKBox** - Another popular dataset is the KKBox dataset, which was released by the WSDM Cup 2018 Challenge (KKBox, 2018). This dataset contains the listening records and the description of the track, including the genre, artist, composer, and lyricist. These datasets vary by size, and the KKBox is the sparsest one.

d. News:

- **MIND** - MIND (Wu et al., 2020) is a recently released large-scale news dataset containing a title, an abstract, a category label, and a body for each news article, as well as interaction records for each user. Moreover, entities in each news article are recognized and mapped into the Wikidata (Vrandečić & Krötzsch, 2014) knowledge base. The entities, their corresponding triplets in Wikidata, as well as entity and relation embeddings learned with the TransE (Bordes et al., 2013) model, are all included in the dataset, which makes it convenient for the research of KG-based news recommendation.

e. Product

- **Amazon Review Dataset** - The most popular dataset is the Amazon Review dataset (McAuley et al., 2015). There are multiple subsets of varied sizes in the Amazon Review dataset, including books, cell phones, clothing, music, electronics, CDs and vinyl, Beauty, etc. Besides the user and item attributes, user reviews and user behavior relations have also been included. Although external knowledge of products is also available, most works (Zhang Y. et al., 2018), (Ai et al., 2018), (Zhao et al., 2017), (Xian et al., 2019), (Zhao et al., 2019) build the user-item KG directly with multiple types of relations within the recommendation dataset.

f. Point of Interest (POI):

- **Yelp** - The most popular dataset is the Yelp Challenge (Yelp, 2013), released in 2013, which contains the attributes of businesses and users, check-ins, and reviews. The Yelp Challenge dataset contains rich POI side information and user-related relations; therefore, the user-item KG can be constructed with knowledge within the dataset.

2.8.4 Summary of the Section

This section covers the multifaceted process of developing XRS models, addressing the intricate challenges and complexities associated with achieving transparency and

comprehensibility in RSs. The exploration in this study encompasses the latest advancements in KG, KGE, and RL techniques, all of which play a crucial role in achieving the overarching objectives of creating RSs that are not only effective but also interpretable. Furthermore, this study conducts a comparative analysis of various works in the field of XR, aligning them with the research objectives and assessing their applicability to specific datasets and technologies under consideration. Additionally, this section provides a list of commonly used datasets for KG-based RSs. Below is the summary of all the XR works discussed in this section.

Table 2.5 Comparison of Pertaining Explainable Recommendation Works

Model	Dataset	KG Usage Type	KGE Method	Explainability Method	Algorithm Method	Approach
TransD-KGAT (Xie, 2023)	commodity knowledge graphs	Embedding	TransD	NA	graph neural network	Build commodity KGs that capture semantically similar and higher-level words, and introduce the TransD-KGAT model, which leverages graph neural network embedding techniques.
BEM (Ye et al., 2019)	FB15k-237 Pagelink Network Alibaba Taobao	Embedding	TransE Node2Vec LINE Doc2Vec Sentence2Vec GraphSAGE	NA	GNN	They introduce a Bayesian framework that combines information from KGs and behaviour graphs. To elaborate, BEM utilizes pre-trained embeddings from the KG and seamlessly incorporates them with pre-trained embeddings from the behaviour graphs through a Bayesian generative model. BEM effectively enhances the embeddings from both sources in a manner that retains their underlying structural characteristics.
KV-MN (Huang et al., 2018)	Last.fm MovieLens ml-20m MovieLens ml-1m Amazon Book	Embedding	TransE	NA	RNN	Combine RNN-based networks with a Key-Value Memory Network (KV-MN) and integrate knowledge base (KB) information to improve the semantic representation of the KV-MN.
CFKG Zhang, Ai, et al. 2018	Amazon	Embedding	TransE	NA	NA	They extend the design philosophy of CF to learn over the KG for personalized recommendation.

to be continued…

… continuation

KTGAN (Yang et al., 2018)	Douban	Embedding	Word2Vec MetaPath2Vec	NA	DNN GAN	The framework incorporates a diverse set of feature embeddings, encompassing user-movie interactions and information extracted from KGs and tags. These representations are simultaneously input into a generator and a discriminator as part of an adversarial training process.
CKE (Zhang F. et al., 2016)	MovieLens-1M IntentBooks	Embedding	TransR	NA	auto-encoders	DL embedding techniques like stacked denoising, and convolutional auto-encoders were employed to extract textual representations for items, while visual representations were extracted using similar techniques. Furthermore, the method represents a cutting-edge neural RS that takes a holistic approach by integrating various data formats. Unlike traditional RSs relying solely on user-item interactions, CKE combines matrix factorization with heterogeneous data sources, including textual content, visual data, and structural knowledge bases. This integration enables the model to derive more informative and context-aware embeddings, facilitating improved accuracy and relevance in top-N recommendations.
ECFKG (Ai et al., 2018)	Amazon	Embedding	TransE	Knowledge-based	Fuzzy-SMA	An approach of joint learning. A soft matching algorithm is proposed to generate personalized explanations for the recommended items
Hete-MF (Yu et al., 2013)	MovieLens-100K IMDb	Meta-Path based similarity	NA	NA	Matrix Fatorization	They suggest constructing a recommendation framework using MF, which takes into account both user ratings and a connected HIN. It extracts L different meta-paths and calculates item-item similarity in each path

to be continued…

… continuation

Hete-CF (Luo et al., 2014)	DBLP	Meta-Path PathSim	NA	NA	NMF & CF	They introduced Hete-CF, an algorithm for social CF that leverages heterogeneous relationships. It finds the user affinity to unrated items by taking the user-user, item-item, and user-item similarity together as regularization terms
HeteRec-p (Yu et al., 2014)	MovieLens-100K IMDb	Meta-Path based	NA	NA	MF Bayesian Ranking	Considers the importance of different meta-paths for different users. "HeteRec-p" first clusters users based on their past behaviours into c groups and generate personalized recommendation with the clustering information instead of applying a global preference model.
FMG (Zhao et al., 2017)	Yelp Amazon Electronics	Meta-Graph based	NA	NA	MF Factorization Machine	Replacing the meta-path with the meta-graph. Since a meta-graph contains richer connectivity information than a meta-path, FMG captures the similarity between entities more accurately
MCRec (Hu et al., 2018)	Movielens LastFM Yelp	Meta-Path based	Path Embedding	NA	CNN	learns the explicit representations of meta-paths to depict the interaction context of user-item pairs.
RKGE (Sun et al., 2018)	MovieLens IMDB Yelp	Meta-Path	Path Embedding	Model-intrinsic	RNN	It autonomously acquires semantic representations of entities as well as the connections between these entities, to describe user preferences for items.
KPRN (Wang et al., 2019)	MovieLens IMDB KKBox	Path-based	Path Embedding	Model-intrinsic	LSTM	Constructs the extracted path sequence with the entity and the relation embeddings and encodes it with an LSTM layer. A sequential RNN model has been developed over the qualified paths between user-item pairs to predict the pair ranking score.
PGPR (Xian et al., 2019)	Amazon	Path-based	TransE	Post-hoc (Guo et al., 2020)	RL MDP & elu, REINFORCE	Combines MDP and RL

to be continued…

… continuation

EKAR (Song et al., 2019)	MovieLens, DBpedia	Path-based	ConvE	Post-hoc	RL MDP, BPR & ReLU	Combines MDP and RL
KGDQN (Xu et al., 2021)	Amazon	Path-based	TransE	Model-intrinsic	RL DQN	Combines KG and RL
RippleNet (Wang H. et al., 2018)	MovieLens-1M Book-Crossing Bing-News	Unified	Preference Propagation	Model-intrinsic	Bayesian Framework	Introduce the concept of preference propagation. Specifically, RippleNet first assigns entities in the KG with initial embeddings. Then it samples ripple sets from the KG.
AKUPM (Tang et al., 2019)	Movielens-1M Book-Crossing	Unified	TransR	Model-intrinsic	Attention	AKUPM first applies TransR for the entity representation. During the propagation process, AKUPM learns the relations between entities with a self-attention layer and propagates the user preference toward different entities with bias. Finally, embeddings from different-order neighbours of interacted items aggregate with the self-attention mechanism to obtain the final user representation.
RCoLM (Li et al., 2019)	Movielens-1M Book-Crossing	Unified	TransR	Model-intrinsic	Transfer Learning	RCoLM trains the KG completion and the recommendation module jointly. It assumes that an item should have a similar latent representation in the two modules. With that assumption, RCoLm unifies two modules and facilitates their mutual enhancement.
KGCN (Wang et al., 2019)	MovieLens-20M Book-Crossing Last.FM	Unified	TransE	NA	GCN	They model the final representation of a candidate item by aggregating the embedding of entities in the KG from distant neighbours of the candidate item. KGCN leverages the idea of GCN by sampling a fixed number of neighbours as the receptive field, which makes the learning process highly efficient and scalable

to be continued…

… continuation

KGAT (Wang et al., 2019)	Amazon-book Last-FM Yelp2018	Unified	TransR	Model-intrinsic	Attention	They model the high-order relations between users and items with embedding propagation. KGAT first applies TransR to obtain the initial representation for entities. Then, it runs the entity propagation from the entity itself outwardly. During the outward propagation process, entity information will interact with the multi-hop neighbours iteratively.
IntentGC (Zhao et al., 2019)	Alibaba Taobao Amazon	Unified	NA	NA	Graph Convolution	They exploit rich user-related behaviours in the graph for a better recommendation. They designed a faster graph-convolutional network (GCN) to guarantee the scalability of IntentGC.
AKGE (Sha et al., 2021)	MovieLens-1M Last-FM Yelp	Unified	Hierarchical Attentive KGE	NA	GNN Attention	AKGE uses an attention-based GNN to propagate the information from neighbours for the final representation of this user-item pair. The construction of the subgraph removes fewer related entities in the graph, facilitating mining high-order user-item relations for the recommendation.
ReMR (Wang et al., 2022)	Amazon	Jointly encodes ontology and instance views	JOIE (Hao et al., 2019)	Post-hoc	RL Actor-Critic RL method,	RL framework for multi-level recommendation reasoning over KGs, which leverages both ontology-view and instance-view KGs to model multi-level user interests
JRL (Zhang Y. et al., 2017)	Amazon	NA	NA	NA	NA	It incorporates multimodal information encompassing images, textual data, and user ratings within a neural network framework. By jointly learning representations from these diverse data sources, JRL is adept at capturing intricate patterns and relationships that exist across multiple modalities, enhancing its ability to generate top-N recommendations that are not only accurate but also tailored to the nuanced preferences and interests of users.

"NA" refers to the non-availability of information.

Table 2.5 compares various XR works discussed throughout the chapter and their respective approaches. These approaches have incorporated KG and their variations in their methodologies for implementing XR models. As depicted in Table 2.5, these approaches have utilized KGs in various forms, such as embedding, path-based, or unified approaches.

The main objective of these XR models is to enhance the explainability of the recommendations they generate. The explainability is generated through either model-intrinsic or model-agnostic approaches. The implementation algorithms employed include GNN, GCN, CNN, RNN, LSTM, Attention-based, Transfer-learning, Bayesian methods, Matrix Factorization, Auto-encoders, GAN, and RL.

These methods leverage data from diverse domains and have demonstrated their effectiveness in improving the explainability and efficiency of the recommendations generated by their models.

2.9 EXPLAINABLE RECOMMENDATIONS – QUANTITATIVE EVALUATION

Most recommenders have been evaluated and ranked on their prediction power, i.e., their ability to predict the user's choices. However, accurate predictions are crucial but insufficient to deploy a good recommendation engine. In many applications, people use an RS for more than exact anticipation of their tastes. Users may also be interested in discovering new items and preserving their privacy, in the fast responses of the system, and in many more properties of the interaction with the recommendation engine. The RSs should support a balance between the explore-exploit ratio. This leads to the necessity of the identification of the right metrics for properly evaluating any RSs (Gunawardana et al., 2012).

2.9.1 Evaluation of Explainability in XR

The advances in KGs, RL, and language models assist in achieving an XAI vision for both the users and the decision-makers. A need arises to validate the explanations generated by recommendation engines.

The quantitative evaluation of the quality of the recommendation explainability is an issue. The quality evaluation of the explanation has always been a qualitative approach. Due to the human orientation, this qualitative approach is more subjective and prone to biases. There is currently no standard for evaluating the explanatory features of RSs, especially from a quantitative standpoint (Rosenfeld, 2021). Thus, there is a need for quantitative evaluation metrics to measure the effectiveness of the explanations to avoid human biases.

To evaluate the performance of XR algorithms, this study can use the same metrics as when analysing traditional recommendation algorithms as discussed in section 2.2.5. For rating prediction tasks, this study can utilise Root Mean Squared Error (RMSE) or Mean Absolute Error (MAE), and for top-n recommendations, it can employ conventional ranking measures like accuracy, recall, HR, F-measure, and NDCG.

However, the best way to assess the explainability of the recommendations is through user studies and online evaluations, which might not always be accessible to researchers due to various resource constraints. In general, there are two approaches to evaluating recommendation explanations offline. The first approach is to evaluate the percentage of recommendations that can be explained by the explanation model, regardless of the explanation quality. The second approach is to evaluate the explanation quality directly.

For the first approach, (Abdollahi & Nasraoui, 2017) adopted Mean Explainability Precision (MEP) and Mean Explainability Recall (MER) metrics. They defined explainability precision (EP) as the proportion of explainable items in the top-n recommendation list relative to the total number of recommended (top-n) items for each user. They also defined explainability recall (ER) as the proportion of explainable items in the top-n recommendation list relative to the total number of explainable items for a given user. Finally, MEP and MER are the averages of EP and ER across all testing users, respectively. This approach is adopted because not all the recommended items can be supported with explanations.

(Peake & Wang, 2018) further generalised the idea and proposed the model Fidelity as a measure to evaluate XR algorithms, which is defined as the percentage of explainable items in the recommended items:

$$Model\ Fidelity = \frac{|explainable\ items \cap recommended\ items|}{|recommended\ items|} \quad \ldots(2.16)$$

For the second approach, evaluating the quality of the explanations usually depends on the type of explanations. One commonly used explanation type is a piece of explanation sentence. In this case, text-based quality evaluation measures assist with offline evaluation. This evaluation approach usually evaluates end users' satisfaction and the usefulness of the given or generated explanations (Mohseni et al., 2021).

The methods for evaluating explainability stated above are qualitative and have the potential for bias. Efforts at quantitative evaluation of XAI can be found in the work of (Rosenfeld, 2021), in which measures are based on the difference between the system's performance using models with higher fidelity and models with lower fidelity, the number of rules in the outputted explanation, the number of nodes used by the system to generate the explanation, and the stability of the system's explanation. The research, however, did not explain the explainability evaluation in the realm of recommendation engines with the framework of KGs and related technologies. Besides that, human interaction is required to evaluate the performance difference between the system's model and the logic presented as an explanation.

2.9.2 Summary of the Section

In this section, this study delves into the evaluation methods employed to assess the effectiveness of XRS and the quality of the generated explanations. This study shed light on the challenges associated with establishing a consensus on the most suitable metrics for evaluating XRS despite the widespread adoption of metrics like NDCG, Recall, HR, and Precision in the research community. While these metrics offer valuable insights into the performance of XRS, this study also acknowledges their

limitations. It explores the need for more comprehensive evaluation approaches that capture both recommendation accuracy and the quality of explanations generated.

2.10 DISCUSSIONS AND RESEARCH GAPS

Exploring the relevant literature surrounding the research objectives revealed the essential components of building an XRS. This study discerned that achieving explainability in RSs necessitates the integration of multiple technologies, including KG, KGE techniques, and a robust method for traversing the KG to generate recommendations with accompanying explanations. The scope of the inquiry was expanded to include the field of RL, where a thorough analysis was conducted on a range of recent works that provided insights into the complex elements involved in attaining explainability within this particular context.

Table 2.5 summarizes an overview of the research conducted in the domain of XRS, with a specific emphasis on the KG-based methods. These efforts have demonstrated notable improvements in the explainability and accuracy of recommendations. Moving forward, **Table 2.6** delves deeper into a critically assessed summary of key advancements and drawbacks of RL-based research in this area, which uses similar datasets intended to be used in this research and identifies the research gaps that are the focal points of this study.

Table 2.6 Existing Research - key Contributions and Limitations

Model	Key Contributions	Limitations
EKAR (Song et al., 2019)	Leverages the power of deep RL within an MDP framework to construct intelligible pathways guided by user preferences.	• Its implementation requires a substantial volume of personalized policy training data to achieve optimal performance. • Notably, the approach suffers from certain limitations, such as the absence of an intermediary reward mechanism, wherein binary rewards (+1, 0, or -1) are assigned based on the fulfilment status of paths involving terminal entities. • Consequently, EKAR faces notable challenges in delivering refined product recommendations and their accompanying explanations, as its decision-making process heavily relies on path probabilities or rewards to determine the prioritization of products for recommendation.

to be continued…

… continuation

KGDQN (Xu et al., 2021)	Integrates KGs into a DQN framework for item recommendations. The DQN uses a type of RL algorithm for solving tasks by learning optimal actions through trial and error.	• The approach inherits drawbacks from the DQN model, including issues of overestimation bias and training instability • The method still faces challenges in determining optimal recommended products and the associated reasoning paths
ECFKG (Ai et al., 2018)	The algorithm conducts breadth-first search of maximum depth from the user-item entities to search for an explanation path that can potentially exist between users and items.	• The main drawback is that sampling paths via breadth-first search are very inefficient and may miss meaningful paths. • Explanations can be provided by finding the shortest path from the user to the recommended item through the KG, which is a kind of post-hoc explainability (Zhang & Chen, 2020).
PGPR (Xian et al., 2019)	Train an RL agent for pathfinding. They formulated the recommendation problem as an MDP to find a reasonable path connecting the user-item pair in the KG. They trained an agent to sample paths between users and items by carefully designing the path-searching algorithm, the transition strategy, terminal conditions, and RL rewards. In the prediction phase, PGPR can generate recommended items for users with specific paths to interpret the reasoning process.	• Nonetheless, the method encounters several challenges, such as high computational demands arising from intricate KG structures, • potential limitations in tailoring recommendations to individual preferences, • the possibility of suboptimal suggestions, • and a potential oversight of relevant factors beyond the sole consideration of path probabilities in the reasoning process.
ReMR (Wang et al., 2022)	Capitalizes on ontologies and instance KGs to effectively represent user interests spanning multiple levels, thereby capturing intricate hierarchical relationships. By integrating multi-level reasoning paths, ReMR significantly enriches the modelling of user preferences, culminating in the generation of enhanced recommendations.	• Overlooking rewarding agents for pursuing intermediate paths aligned with user preferences • quantitative evaluation of explainability • recommending the prioritized products
RKGR-RNS (Zhang et al., 2022)	The research focus on the reward mechanism in RL for XR and introduce a negative sampling method into the RL model to effectively delineate user preferences	• There is no mechanism to address the mis-tagging of positive samples as negative samples, which affects the overall sentiment and accuracy of the model.
HR-RLKG (Song et al., 2023)	The research introduces a method that utilizes RL to enable agents to traverse diverse paths more effectively and leverage the underlying KG by employing the TransD embedding structure	• The choice of embedding method has a significant impact on recommendation performance.

Path-based recommendation methods, such as PGPR (Xian et al., 2019), ADAC (Zhao et al., 2020), KGDQN (Xu et al., 2021), EKar (Song et al., 2019), ReMR (Wang et al., 2022), RKGR-RNS (Zhang et al., 2022), and HR-RLKG (Song et al., 2023), employ RL as a MDP to overcome the challenge of explaining entities that are more than one-hop away by searching for feasible paths between user-item pairs in the KG (Chen et al., 2023).

(Xian et al., 2019) introduced PGPR, a method combining MDP and RL to navigate multi-hop paths in the KG for offering recommendations. While PGPR ensures transparency by providing connectivity paths, it faces computational costs, personalization, optimal product suggestions, and their associated reasoning challenges. Notably, the approach prioritizes reasoning path selection solely based on path probability, overlooking other factors.

In a subsequent study, (Xu et al., 2021) developed KGDQN by integrating KGs into a deep Q-network (DQN) for item recommendations. However, it inherits limitations from DQN, such as overestimation bias and instability. While Xu introduced a non-binary rewards strategy, it still grapples with issues concerning optimal products and their associated reasoning paths, determined by maximizing the multiplication of rewards with corresponding probabilities.

(Song et al., 2019) introduced EKAR, which uses deep RL and MDP to create understandable paths based on user preferences. However, this approach requires a significant amount of personalized policy training data. Additionally, it lacks an intermediate reward mechanism and may receive binary rewards (+1, 0, or -1) depending on the path completion state with the terminal entity. This method faces challenges in suggesting optimal recommended products and their corresponding reasoning, as it primarily relies on path probabilities or rewards to prioritize products.

(Wang et al., 2022) proposed ReMR, a method leveraging ontology and instance KGs to model multi-level user interests, enhancing the modeling of user interests and leading to improved recommendations.

(Zhang et al., 2022) introduced RKGR-RNS, focusing on the reward mechanism in RL for XRS and introducing a negative sampling method into the RL model to effectively delineate user preferences.

(Song et al., 2023) proposed HR-RL-KG, a method that utilizes RL to enable agents to traverse diverse paths more effectively and leverage the underlying KG by employing the TransD embedding structure.

In summary, each model targets different aspects of explainability, user preference adaptation, and knowledge representation to suit varied application needs. However, challenges remain regarding personalization, optimal product recommendations, and identifying effective reasoning paths for these recommendations. Most RL-based approaches fail to reward agents for exploring intermediary sub-paths that are subsets of guided path patterns and may align with user preferences. Moreover, discovering meaningful traversal paths on large graphs remains a challenge, impacting the overall accuracy of recommendations.

Indeed, while recent advancements have undeniably advanced RSs and their transparency, they continue to grapple with several persistent challenges. One of these challenges pertains to the appropriate form of explainability, which should ideally be intrinsic to the model itself and seamlessly integrated within the RS. This intrinsic explainability should be rooted in the genuine reasoning process that guides the model's recommendations to users. The correctness of this explainability fosters trust among users and directly contributes to the enhanced performance of RSs.

It's worth noting that many of the RL-based approaches mentioned earlier in **Table 2.6** have limitations when offering the right form of explainability. In some instances, such as with ECFKG, the explainability is added after the fact in a post-hoc manner. In other cases, the process of generating explainability is constrained by suboptimal choices of traversal paths, which fail to strike the right balance between exploit and explore strategies, among other considerations. These challenges underscore the ongoing pursuit for more effective and meaningful explainability within RSs, a vital element in ensuring their continued evolution and user acceptance.

Based on the observed gaps in existing research, this study focuses on the following research objectives:

- Notably, the observations made during the study have uncovered a recurring issue in the research landscape. Specifically, there is a notable oversight in many studies where agents are not sufficiently incentivized to explore intermediate paths that align with user preferences. This oversight poses challenges when attempting to provide personalized recommendations that truly cater to individual users' unique needs and preferences. This observation underscores the significance of the primary research objective, as outlined in Chapter 1 (RO1), which aims to address and rectify this issue, ultimately striving to enhance the quality of personalized recommendations within RSs.

- The second research objective, as highlighted in Chapter 1 (RO2), is to determine the most effective traversal path while considering the balance between exploration and exploitation. This particular area currently constitutes a significant research void within the field. Finding the optimal traversal path has the potential to enhance explainability and boost the performance of the RS, ultimately instilling greater trust in users regarding the recommendation process.

- The third research objective, as delineated in Chapter 1 (RO3), is to propose recommendations and prioritize products to augment the overall performance of the RS. This objective underscores the significance of refining the product recommendation process to enhance the efficacy and efficiency of the RS.

- The study identified a crucial gap in existing research related to RSs and their explainability. Most current approaches tend to rely on qualitative assessments to gauge the effectiveness of explainability, lacking a quantitative mechanism to evaluate it within the RS domain. Recognizing this gap, the research study has prioritized a critical fourth objective, as delineated in Chapter 1 (RO4), to introduce a quantitative mechanism for measuring the quality of explainability. By addressing this objective, the study aims to contribute significantly to the field by

providing a more robust and objective means of assessing the explainability of RSs, ultimately enhancing their transparency and usability.

The proposed approach aims to address the challenges posed by XRS by focusing on identified research objectives. The first three objectives revolve around enhancing the performance of XRS, setting a benchmark by systematically comparing them to state-of-the-art baseline models. In addition, the fourth objective aims to bridge a critical research gap by introducing a robust quantitative measure for assessing the quality of explainability in RSs. Through these objectives, this research aims to contribute significantly to the ongoing evolution of XRSs, ultimately advancing their effectiveness and usability in real-world applications.

2.11 SUMMARY

This chapter offers an extensive exploration of the existing literature, delving deeply into various facets of XRS. It comprehensively examines the realms of XAI, RSs, KG, KGE, and RL. The review meticulously analyses recent research aligned with the specific research objectives, shedding light on the state of the field, and identifying key areas where this research is poised to make a significant impact. Furthermore, it identifies critical research gaps, emphasizing the pressing need for the study and underscoring its relevance in advancing the domain of XRS.

CHAPTER III

RESEARCH METHODOLOGY

3.1 INTRODUCTION

This chapter outlines the research methodology used in this study, providing a structured framework for systematically addressing the research problem. The methodology includes the general research framework and the necessary steps and phases for conducting the research. As such, selecting an appropriate methodology is crucial for achieving the research objectives.

At the core of this effort is the improvement of RS performance by leveraging semantic information about entities, their attributes, and the complex relationships between them. Providing clear and understandable explanations for the recommendations generated is also essential. Another goal is to develop a quantitative measure to accurately assess the effectiveness of these explanations from the end users' perspective.

This study is structured into crucial stages, each specifically designed to achieve the primary goal of the research. These carefully planned stages provide a systematic approach to addressing the research questions and ultimately achieving the stated research objectives.

The chapter follows this structure: Section 3.2 explains the research design strategy, outlining its general flow. The operational framework is presented in Section 3.3, detailing the activities involved within each phase of the study, ranging from the literature review to the final report writing, and the further exploration covered in detail in sections 3.4 to 3.10. Finally, the chapter is summarized in section 3.11.

3.2 RESEARCH DESIGN STRATEGY

This section outlines a comprehensive research strategy adopted to achieve the research objectives identified in this thesis.

This research aims to improve the performance of RS by using semantic information about entities, their attributes, and relationships. The goal is to provide clear and understandable explanations for the recommendations generated and to develop a quantitative measure to assess the effectiveness of these explanations from the end users' perspective.

To achieve that, this research reviewed recent studies to find gaps and identified research questions, hypotheses, and objectives. This research put efforts into finding the research gaps aligning with research objectives. Later, this study identified relevant data and recent studies in the field aligning with research objectives. This research then built and tested recommendation models with this data to achieve the research objectives. This research also used advanced explainable techniques to explain the recommendations generated by the developed model. This research compared the effectiveness of the XRS model with other baseline studies. Overall, this research assists both RS and XAI fields by using semantic information to make recommendations better and easier to understand.

This study implemented the operational framework of the research methodology outlined below as part of its research design strategy.

3.3 RESEARCH METHODOLOGY – OPERTATIONAL FRAMEWORK

The operational framework is the expansion of the research design. It comprises six distinct phases ranging from the literature review to the final report writing and compilation, each delineating a crucial aspect of the research process. It outlines a structured guide for conducting each phase of the research.

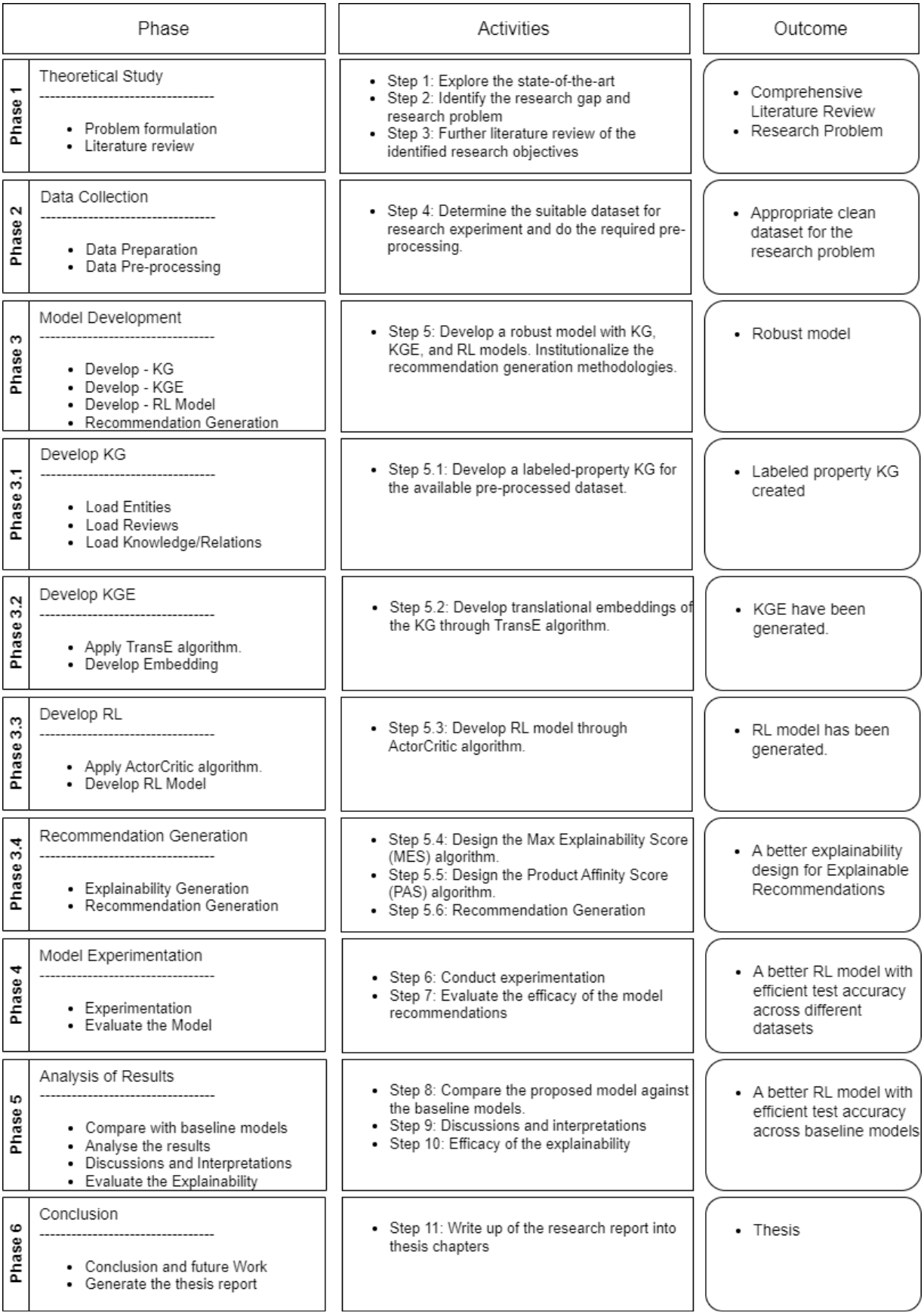

Figure 3.1 Research Methodology - Operational Framework

Figure 3.1 illustrates the operational framework of this research study, composed of seven phases. The first phase entails an in-depth exploration of existing literature and formulates the research problem to establish a comprehensive theoretical foundation. Phase two focuses on preparing and preprocessing the experimental data to ensure its suitability for analysis. Phase three is dedicated to building a strong XRS model through the creation of KG, KGE, and RL models. This phase comprises four sub-phases. Phase 3.1 focuses on constructing a semantic structure for the dataset through labelled property KG. Phase 3.2 covers the embedding construction of the designed semantic graph structure using advanced techniques to address the inherent issues of KG. Phase 3.3 develops an RL model to enhance the prediction accuracy, thus achieving the first objective of this study. Phase 3.4 focuses on the development of recommendation generation and their corresponding explainabilities. The subsequent phase involves model experimentation to assess the model's efficacy. In the fifth phase, a comparison is drawn between the proposed approach and state-of-the-art baseline models to validate its effectiveness, thereby responding to specific research objectives. The sixth phase encompasses the synthesis of all findings into a comprehensive research report. This systematic framework effectively guides the entire research endeavour, ensuring a comprehensive and systematic progression from initial problem formulation to conclusive reporting.

The following sections delve deeply into each phase of the operational framework of the research methodology.

3.4 PHASE 1: THEORETICAL STUDY

The theoretical study is the initial phase of the research and aims to provide a concise understanding of the concepts, theories, gaps, and techniques relevant to the research topic. It encompasses literature review and problem formulation.

3.4.1 Literature Review

A literature review serves as a crucial phase in research, entailing a systematic exploration of a specific research area. It aims to establish a coherent understanding of

the subject matter by defining key concepts, surveying prevailing techniques for addressing pertinent challenges, and identifying existing gaps that require further resolution. This process involves an exhaustive examination of relevant topics spanning different periods.

The review encompasses seven principal focus areas: RSs, XAI, KGs, KGE, RL, XRS and explainability evaluation mechanisms. These focus areas are investigated thoroughly, delving into core principles, methodological approaches, and the prevailing hurdles within each focus area. Moreover, the review extends to probing questions intrinsic to these topics. For instance, it seeks solutions to existing challenges within each field and determines optimal methods for enhancing the performance of specific applications.

The culmination of this literature review lies in pinpointing the voids present within the current body of work. This critical analysis lays the groundwork for selecting appropriate techniques that can be integrated into the proposed frameworks. By discerning these gaps, this study identifies the problem statement based on the research gap and offers fresh perspectives and advanced solutions that contribute significantly to the advancement of the field of XR. The literature review process is a dynamic and analytical endeavour that serves to summarize existing knowledge and shape the trajectory of future research endeavours.

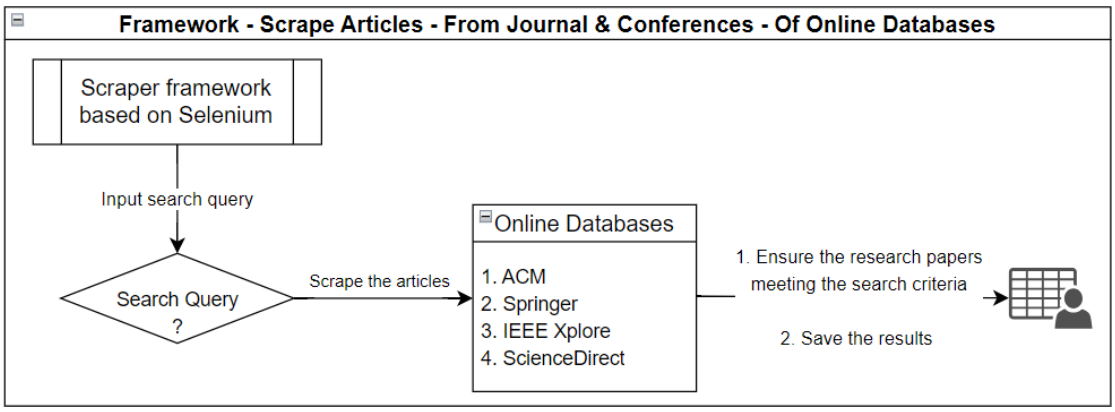

Figure 3.2 Web Scraper Framework

This study introduces a custom web scraper, as depicted in **Figure 3.2**, to simplify accessing and organizing research papers from premier online databases like ACM, Springer, IEEE Xplore, and ScienceDirect. Built with Python, the scraper uses the Microsoft EDGE web driver and Selenium for automated web scraping. It helps the study efficiently find and manage relevant academic content. The custom web scraper framework is available on GitHub link[1].

3.4.2 Problem Formulation

Leveraging insights from an extensive literature review in the field of XR, as detailed in **Chapter II**, this study has effectively pinpointed significant research gaps and formulated a concise problem statement. At the heart of this challenge lies the development of a robust XRS. This system not only endeavours to offer user recommendations tailored to their preferences but also endeavours to provide thorough reasoning or explanations for these recommendations. Furthermore, a deficiency highlighted in existing literature concerns the lack of a quantitative metric to evaluate the quality of explainability.

The identified issues have formed the basis for crafting research questions, hypotheses, and objectives, as elaborated in **Chapter I**, which serve as guiding principles for this study. These research questions, in turn, are instrumental in shaping the research hypotheses. Together, the research questions and hypotheses contribute to defining the primary research objectives. This study has outlined four pivotal research objectives, all focused on the development of a robust XRS model and the proposal of a metric capable of quantitatively assessing the quality of the explanations generated.

3.5 PHASE 2: DATA COLLECTION

Transitioning to the subsequent phase of this study, the spotlight shifts to the data collection process, a pivotal stage within the research. Data collection is a critical activity in any research. It selected a suitable data set for evaluating the performance of the proposed models in the study. After the dataset is selected, it goes through a preprocessing step to transform the raw data into a form that can be accurately analysed. This phase contains two processes: data preparation and data preprocessing.

1. https://github.com/neeraj-tiwary/PhD-ScrapeResearchArticles

3.5.1 Data Preparation

a. Datasets

Four distinct e-commerce domain-specific datasets have been selected from Amazon for experimental research. Specifically, these datasets pertain to the CDs & Vinyl, Clothing, Cell Phones, and Beauty (He & McAuley, 2016), (McAuley & Leskovec, 2013). Each dataset comprises 13 distinct zipped text files containing information about entities, their relationships, and user transactions for purchasing products. **Table 3.1** outlines the descriptions of these text files, while **Table 3.2** presents the data sizes of these text files for the various datasets utilized in the experiment.

Table 3.1 Description of Dataset Text Files Utilized in the Experiment.

Text files	Description
users.txt.gz	List of masked users
products.txt.gz	List of masked products to be recommended to users
related_product.txt.gz	List of masked related bought/viewed products
category.txt.gz	List of categories of the products
brand.txt.gz	List of brands or manufacturers of the products
vocab.txt.gz	List of various products described words from reviews
brand_p_b.txt.gz	An association of products to corresponding brands
category_p_c.txt.gz	An association of products to corresponding categories
also_bought_p_p.txt.gz	An association of related products also bought with purchased products
also_viewed_p_p.txt.gz	An association of related products also viewed with purchased products
bought_together_p_p.txt.gz	An association of related products bought together with purchased products
train.txt.gz	A list of users' purchased products and their described feature words (used for model training)
test.txt.gz	A list of users' purchased products and their described feature words (used for model validation)

Table 3.2 The Size of the Dataset Text Files Utilized in the Experiment.

Text files \ Sizes	CDs&Vinyl (in KB)	Clothing (in KB)	Cell Phones (in KB)	Beauty (in KB)
users.txt.gz	700	367	260	208
products.txt.gz	265	122	56	63
related_product.txt.gz	1,144	1,795	531	871
category.txt.gz	5	8	2	2
brand.txt.gz	10	8	6	14
vocab.txt.gz	876	79	84	85
brand_p_b.txt.gz	17	10	9	18
category_p_c.txt.gz	307	97	12	22
also_bought_p_p.txt.gz	9,397	3,715	1,409	2,163
also_viewed_p_p.txt.gz	53	359	32	359
bought_together_p_p.txt.gz	133	43	19	23
train.txt.gz	260,866	22,104	22,640	21,804
test.txt.gz	100,467	6,811	6,997	7,404

b. Data Snapshot

Table 3.3 provides a comprehensive snapshot of the Amazon Beauty e-commerce dataset employed in our experimentation. The upper segment furnishes snapshot information concerning entities within the Beauty dataset, utilizing masked information for the User, Product, and Related Product entities. In the lower section of Table 3.3, the leftmost column highlights the three most crucial relationships within the dataset. For instance, User #319, identified in the User entity, purchases Product #8099 from the Product entity, establishing a relationship denoted as “user purchasing product”. Simultaneously, User #319 has mentioned various words, including #10578 and 13241, documented in the Word entity during the purchase of Product #8099. Similarly, Product 8099 is described by distinct words like #10578, 13241, 18022, 19428, etc., recorded in the Word entity, creating a relationship denoted as “product described as words”.

Subsequent columns detail relationships between products and other entities, such as Brand, Category, and Related Product. Each row corresponds to a sequential product number. For instance, Product #1 in the Product entity is produced by Brand #8 from the Brand entity, categorized under Category #0, 20, 18, 19, and 4 in the Category entity. Additionally, Product #1 lacks connections in the “Also Bought” and “Bought Together” categories with related products but is linked through “Also Viewed” relationships with Related Products #7644, 7647, and numerous others.

This structured representation underscores the intricate network of associations in the Amazon Beauty e-commerce dataset, highlighting the interplay between entities and relationships. Importantly, entities serve as dimensions, while relationships assume the role of factual connections between these dimensions.

Table 3.3 Sample Snapshots (Top 5 Records Only) From the Amazon Beauty e-commerce Dataset

Entity -Dataset snapshot					
Product	User	Word	Brand	Category	Related Product
B003QLRO7W	A67861C2ZZDHQ		COKA	Beauty	B00KR26VFE
B000TBZHNK	A2L6LGTCAO02FY	fawn	Xtreme Brite	Makeup	B00E7LQHZ0
B00613KK96	A33VMF6NERLK3E	sowell	Prada	Face	B00BMW24TU
B000C2359E	A7DYBDGVB2LC4	hanging	Versace	Concealers & Neutralizers	B00K67AQN8
B001JQLNMI	AOP43JFBAILRJ	woody	Avalon Organics	Hair Care	B008GOR6O0

Relationships (including transactional relations from - Train & Test) -Dataset snapshot							
User Purchase Product, User Mention Word, Product Described As Word			Product Produced Brand	Product Belongs To Category	Product Also Bought Related-Product	Product Also Viewed Related-Product	Product Bought-Together Related-Product
User	Product	Word	Brand	Category	Related-Product	Related-Product	Related-Product
319	8099	10578 13241 …	8	0 20 18 19 4		7644 7647 8868 …	
9816	8099	18022 19428 …	738	0 100 166 46	40766 16262 34704 …	12789 12805 34704 …	
10232	8099	7152 1833 …		0 24 25 29 15	5954 23190 5995 …	5995 5954 7777 …	5954
3222	8099	17730 7462 …	433	0 8 9 7	4184 349 4155 …	4184 380 390 ….	4184
1449	8099	5660 20584 …	30	0 32 2 15	4840 4809 10948 ...	10921 4840 4803 …	4840

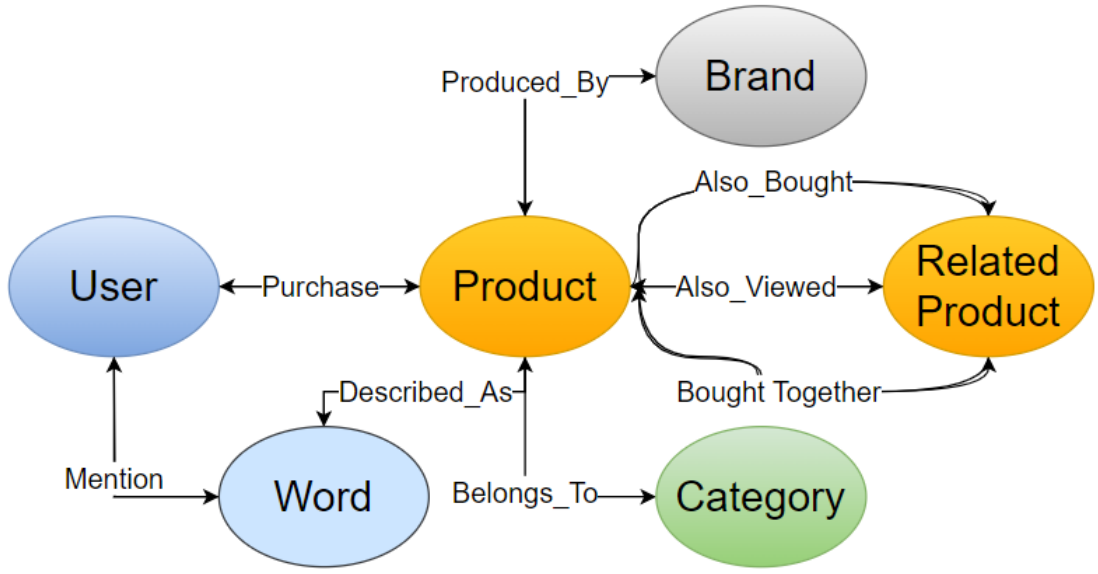

Figure 3.3 **Beauty - Conceptual Relationship Among Entities**

c. Data Relationships

These datasets consist of six entities: User, Product, Product's feature words, Related Products, Brand, and Category. The relationships among these entities are diverse, comprising eight different types, as illustrated in **Figure 3.3**, forming the basis of KG creation for the respective datasets. These datasets establish connections between users and various products within specific categories and brands. Additionally, products have associations with related products, manifesting as linkages like 'also viewed' or 'bought together'.

This diagram exhibits the conceptual relationships among entities within the Beauty dataset considered in this experiment. The dataset at hand encompasses a user-centric KG composed of multiple entities, including users, products, categories, brands, feature words, and related products. This diagram exhibits the conceptual relationships among entities within the Beauty dataset considered in this experiment. The dataset at hand encompasses a user-centric KG composed of multiple entities, including users, products, categories, brands, feature words, and related products. Users may have a connection to various product purchases within specific categories and brands. Additionally, products demonstrate associations with related products, like those viewed or bought together. These connections illustrate user interactions

and preferences. The dataset captures direct interactions and the relationships between products and feature words mentioned by users. The KG structure facilitates the exploration of head entities (e.g., users), relationships, and their corresponding tail entities (e.g., products). The diverse relation types, including inverse relationships, further enrich these interactions and associations within the data. This framework provides a foundation for uncovering insights into user behaviours, product correlations, and purchase patterns.

d. Reasons for Opting for Amazon Datasets:

The Amazon e-commerce datasets (He & McAuley, 2016) used in this study are meticulously curated and encompass all the necessary attributes for the seamless execution of experimental procedures. Specifically, these datasets encapsulate the historical records of product purchases made by diverse users. Each dataset is considered as an individual benchmark that constitutes a user-centric KG with several types of relations (including inverse relations), which implies that results are not necessarily comparable across different domains.

The wealth of information within these datasets extends beyond mere transactions and holds the key for constructing a comprehensive KG enriched with semantic connections derived from the data. This KG, in turn, serves as the foundation for generating embeddings, intricate numerical representations that succinctly capture the underlying relationships and affinities present within the dataset. These embeddings are extremely helpful in designing the recommendation model using the RL framework, which recommends new products to users and generates the associated explainabilities.

These datasets were used in many previous experiments conducted in this space, which include (Xian et al., 2019), (Song et al., 2019), (Wang et al., 2022), (Ai et al., 2018), and (Zhang Y. et al., 2017). Effectiveness metrics, borne from these datasets and the constructed model, serve as the benchmark for evaluating the performance of the Actor-Critic framework. Through these metrics, a comparative analysis is facilitated, pitting the outcomes of this approach against the backdrop of previous baseline research endeavours. These metrics play a profound role in attaining

the overarching goals of the research. Research objectives are meticulously outlined, and their validation or refinement is found through the lens of these metrics. The effectiveness metrics derived from these foundations holistically gauge the model's performance, driving comparative analysis and providing insights that address the core questions of the research.

e. Data Summary

Table 3.4 presents an in-depth statistical summary of the datasets used in the experimentation, shedding light on the entities, relationships, and distinctive features encapsulated within the data.

The upper segment of the table prominently showcases the quantity of entities contained within each dataset. A notable observation is that the CDs & Vinyl dataset stands out due to its notably larger size when compared to the other datasets. This size discrepancy could potentially influence the findings and analyses conducted on this dataset, underscoring its significance within the research.

The lower segment of the table explores the relationships between head and tail entities in each dataset. It provides insights into the nature of these relationships by presenting mean and standard deviation values. What's noteworthy is that the mentioned relationships are consistently prevalent across all datasets, underscoring their universal importance in capturing crucial product attributes. However, some relationships might include redundant words, potentially impacting their effectiveness. The research proposes employing the Term Frequency - Inverse Document Frequency (TF-IDF) method to address the concerns of data sparsity. This technique can be leveraged to eliminate less pertinent words, ensuring the extraction of relevant information while eliminating noise.

Table 3.4 Descriptions and Statistics of Four Amazon e-commerce Datasets

		CDs & Vinyl	Clothing	Cell Phones	Beauty
Entities	**Description**	**Number of Entities**			
User	User in RS	75,258	39,387	27,879	22,363
Product	Product to be recommended to users	64,443	23,033	10,429	12,101
Feature/Word	A product feature word from reviews	202,959	21,366	22,493	22,564
RelatedProduct	Related bought/viewed products	236,255	339,367	101,287	164,721
Brand	Brand or manufacturer of the product	1,414	1,182	955	2,077
Category	Category of the product	770	1,193	206	248
Relations	**Description**	**Statistics (Mean ± Standard Deviation) of Relations per Head Entity**			
Purchase	*User* $\xrightarrow{purchase}$ *Product*	14.6 ± 39.1	7.1 ± 3.6	6.9 ± 4.5	8.9 ± 8.1
Mention	*User* $\xrightarrow{mention}$ *Feature/Word*	2,545.9 ± 10,942.3	440.2 ± 452.4	652.1 ± 1335.8	806.9 ± 1344.1
Described_as	*Product* $\xrightarrow{described\_as}$ *Feature/Word*	2,973.2 ± 5,490.9	752.7 ± 909.4	1,743.2 ± 3,482.8	1,491.2 ± 2,553.9
Belong_to	*Product* $\xrightarrow{belong\_to}$ *Category*	7.2 ± 3.1	6.7 ± 2.1	3.5 ± 1.1	4.1 ± 0.7
Produced_by	*Product* $\xrightarrow{produced\_by}$ *Brand*	0.2 ± 0.4	0.2 ± 0.4	0.5 ± 0.5	0.8 ± 0.4
Also_bought	*Product* $\xrightarrow{also\_bought}$ *RelatedProduct*	57.3 ± 39.2	61.4 ± 33.0	56.5 ± 35.8	73.7 ± 30.7
Also_viewed	*Product* $\xrightarrow{also\_viewed}$ *RelatedProduct*	0.3 ± 1.9	6.3 ± 6.2	1.2 ± 4.3	12.8 ± 9.0
Bought_together	*Product* $\xrightarrow{bought\_together}$ *RelatedProduct*	0.7 ± 0.8	0.7 ± 0.9	0.8 ± 0.8	0.7 ± 0.7

Particularly intriguing is the examination of relationships between the Product and Related Products datasets. Among the various relationships explored, the "Also

Bought" relationship surfaces as the most prominent. This revelation holds significance, implying that the "Also Bought" link between products is a prevailing indicator of user preferences and interactions.

3.5.2 Data Preprocessing

a. Data Readiness

Ensuring the readiness of experimental data is crucial for any study, and this research has meticulously prepared it, encompassing entities and their relationships. During this data readiness step, the study creates a dataset by compiling the relevant information from the corresponding files, as shown in **Table 3.1**, of entities, their relations, and the transactional information stored in the train and test files.

```
class AmazonDataset(object):
    """This class is used to load data files and save in the instance."""

    Neeraj Tiwary
    def __init__(self, data_dir, set_name='train', word_sampling_rate=1e-4):
        self.data_dir = data_dir
        if not self.data_dir.endswith('/'):
            self.data_dir += '/'
        self.review_file = set_name + '.txt.gz'
        self.load_entities()
        self.load_product_relations()
        self.load_reviews()
```

Figure 3.4 A Code Snippet of Forming the Readiness of the Dataset

This research ensures data readiness by loading all relevant entities from entity files and their associated relationships from relationship files. Additionally, it includes review information from the train and test data files, as illustrated in **Figure 3.4.** The train and test files hold essential information regarding users' product purchases and the corresponding product descriptions. This step involves extracting critical information, such as which products users purchased, and which product descriptions were mentioned by users or described as features of products. This extraction

process is carried out from the vast train and test files. By combining all dataset files, the data becomes ready for further experimentation.

b. Data Relevancy

The subsequent step involves the vital task of data preprocessing. Since the dataset contains numerous product purchases with many feature words mentioned, it presents an issue of data sparsity. This situation includes a long tail of products with less significant feature words or some products with common feature words. These issues can affect data quality and subsequent steps during KG creation. Thus, extracting only relevant feature words and utilising them in the KG creation is important. This step entails extracting pertinent feature words from user-product interactions in the e-commerce dataset. This preparatory step is integral, and a filtering criterion is employed for each dataset. This criterion retains feature words with frequencies below 5,000 to avoid common words and allows those with a TF-IDF score to surpass 0.2, ensuring the inclusion of more relevant feature words. This meticulous preprocessing ensures the preservation of noteworthy and pertinent feature words, thereby elevating the overall quality and relevance of the dataset for ensuing analysis and modelling.

The rationale behind choosing a frequency threshold of 5000 and a TF-IDF threshold of 0.2 is to strike a balance between removing very common and less important feature words while allowing for the inclusion of more uncommon and important words during KG creation. This approach aims to ensure that the extracted feature words are both relevant and informative, thereby enhancing the quality and effectiveness of the subsequent steps in the process.

3.6 PHASE 3: MODEL DEVELOPMENT

Building on the insights gained from the theoretical study, the third phase of this research outlines a comprehensive model development architecture. This architecture bridges the gap between the solution's implementation and the earlier identified issues.

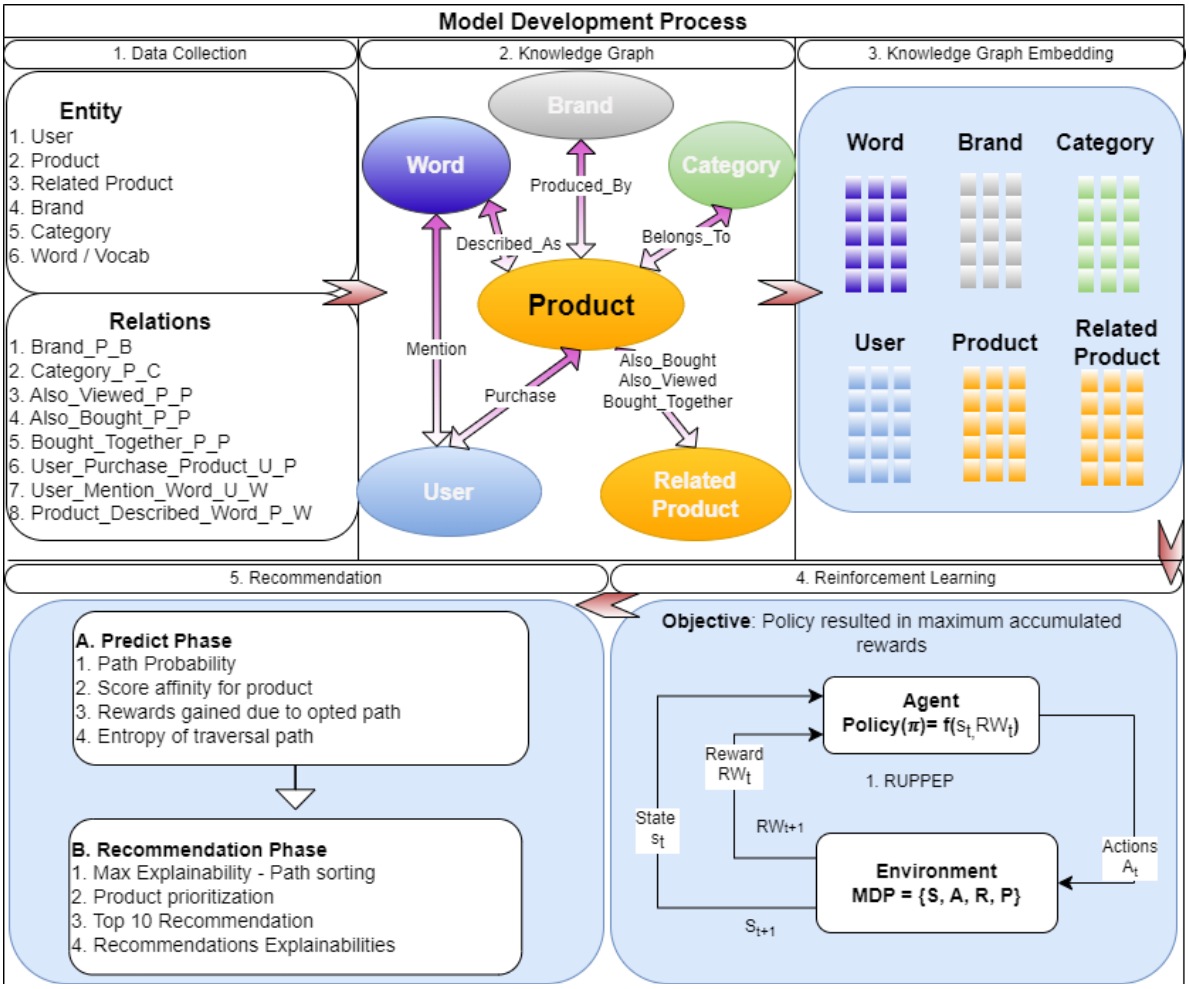

Figure 3.5 Model Development Process

Initiating the construction of this framework involves the creation of a KG, utilizing the collected data. Within this construct, entities embody users and products, while relationships mirror interactions or associations connecting them. The KG can be fashioned using the RDF format or the label property graph (LPG). In this endeavour, an LPG incarnation of the KG was crafted, as depicted in **Figure 3.5 (2)**.

After the KG's formation, the focus shifts to the creation of KGEs. This phase aims to transform entities and relationships into vectorized representations. The "TransE" algorithm, selected as an appropriate embedding technique, effectively translates the KG into its vectorized manifestation, as depicted in **Figure 3.5 (3)**.

Concluding the architectural blueprint is the design and execution of an Actor-critic RL model. This model is ingeniously constructed, with the embedded KG

serving as the environment within which the RL agent operates. The model's training regimen is geared towards equipping it with the ability to navigate and comprehend connectivity pathways linking users and products, as depicted in **Figure 3.5 (4)**.

3.6.1 Develop KG

This study leverages four Amazon e-commerce datasets, each rich with information on products, reviews, categories, users, and other valuable data. Extracting entities and relationships is identified as a pivotal task within this process. Entities represent distinct objects, such as products and users, while relationships define the connections between these entities. For example, a "product" entity may be linked with a "category" entity or multiple "reviews" entities. This extraction phase establishes the fundamental structure of the KG.

a. Ontological Schema

This study defines the ontological schema of the KG in key-value pairs using JSON format, as shown in **Figure 3.6,** for their respective datasets. In this representation, the outer keys correspond to the SUBJECT nodes, the inner keys of the outer values represent the PREDICATE, and the inner values represent the OBJECT nodes in the Subject-Predicate-Object (SPO) triplet within the KG.

```
# Defining the KG relation
KG_RELATION = {
    USER: {
        PURCHASE: PRODUCT,
        MENTION: WORD,
    },
    WORD: {
        MENTION: USER,
        DESCRIBED_AS: PRODUCT,
    },
    PRODUCT: {
        PURCHASE: USER,
        DESCRIBED_AS: WORD,
        PRODUCED_BY: BRAND,
        BELONG_TO: CATEGORY,
        ALSO_BOUGHT: RPRODUCT,
        ALSO_VIEWED: RPRODUCT,
        BOUGHT_TOGETHER: RPRODUCT,
    },
    BRAND: {
        PRODUCED_BY: PRODUCT,
    },
    CATEGORY: {
        BELONG_TO: PRODUCT,
    },
    RPRODUCT: {
        ALSO_BOUGHT: PRODUCT,
        ALSO_VIEWED: PRODUCT,
        BOUGHT_TOGETHER: PRODUCT,
    }
}
```

Figure 3.6 Defining KG-Relation – An Ontological Schema

b. KG Creation - Algorithm

Based on the ontological schema, the study follows the steps outlined in **Figure 3.7**, to develop a KG. The entity-level properties are defined and the relationships between entities are identified. RSs utilize the information available in the KG to enhance recommendation explainability.

```
class KnowledgeGraph(object):

    Neeraj Tiwary
    def __init__(self, dataset):
        self.G = dict()
        self._load_entities(dataset)
        self._load_reviews(dataset)
        self._load_knowledge(dataset)
        self._clean()
```

Figure 3.7 Knowledge Graph Development

1. Entities Identification

Initially, the study creates the entity nodes from the available entities in the dataset.

2. Reviews Identification

Subsequently, it creates the SPO edges for the review data from the dataset, ensuring that it meets the requirements of the frequency threshold and the TF-IDF threshold, as mentioned in section 3.5.2.b.

3. Knowledge Identification

Later, it loads the knowledge information from the relation categories such as [PRODUCED_BY, BELONG_TO, ALSO_BOUGHT, ALSO_VIEWED, BOUGHT_TOGETHER], and creates the corresponding SPO edges.

4. Remove Duplicates

Finally, it drops any duplicate edges from the KG, to ensure the KG's integrity and compute the KG's degree.

3.6.2 Develop KGE

KGs offer valuable insights and recommendations, but their intricate graph structures can pose challenges in terms of navigation and traversal. One approach to mitigating this complexity is through the embedding of KG models. Embeddings are essentially compact numerical representations that capture the semantic essence of entities and

relationships. These representations retain the underlying meaning of the KG elements while condensing them into a format more amenable to computational operations.

KGE models play a crucial role in simplifying the representation of entities and relationships within the KG by mapping them into a lower-dimensional space and enable efficient pathfinding and reasoning between these entities and relationships.

This study employs the Translational Algorithm (TransE) for generating embeddings. TransE is a popular KGE algorithm that operates on the principle of translation. It assumes that a valid relationship between two entities can be represented as a translation from the embedding of one entity to the embedding of another entity via the relationship's embedding (Bordes et al., 2013).

The TransE algorithm is particularly effective in simplifying the complex graph structure of KGs. By mapping entities and relationships to a lower-dimensional space, TransE creates a more organized representation that preserves semantic relationships and facilitates faster calculations of entity connections. The algorithm adjusts the embeddings during training to ensure that the translations between entities and relationships in the embedding space match the actual relationships in the KG.

a. KGE Model – Parameters

In the development of the KGE model, parameters, as shown in **Table 3.5**, play a crucial role in creating a more robust embedding of the KG structures. The key parameters include "epochs", "batch-size", "learning-rate (lr)", "weight_decay", "l2_lambda", "max_grad_norm", "embed_size", "num_neg_samples", and "steps_per_checkpoint".

The delicate balance among these parameters ensures that the developed KGE accurately represents the true semantic essence of entities and relationships. A larger "batch-size" can impact the embedding's quality, while a narrower "steps_per_checkpoint" affects the time required to complete the KGE generation.

Table 3.5 KGE Generation – Parameters Selection

Parameters/arguments	Descriptions	Default Value
dataset	One of {beauty, cd, cell, cloth}	BEAUTY
name	model name	train_transe_model
epochs	number of epochs to train	30
batch_size	batch size	64
lr	learning rate	0.5
weight_decay	weight decay for adam	0
l2_lambda	l2 lambda	0
embed_size	knowledge embedding size	100
steps_per_checkpoint	Number of steps for checkpoint	10000
checkpoint_folder	Checkpoint folder name	checkpoint
is_resume_from_checkpoint	If resume the run from the last checkpoint state?	1
max_grad_norm	Clipping gradient	5.0
num_neg_samples	number of negative samples	5

b. KGE Model - Algorithm

Below is the algorithm followed for generating the embeddings of the KG structure.

1. **Initialize Parameters**: Set the parameters as mentioned in section 3.6.2.a.

2. **Load Data**: Load the KG data, including entities, relationships, and transactional information.

3. **Epochs**: Iterate through the following steps from step 4 to step 8 for the specified number of epochs.

4. **Train Model**: Train the TransE embedding model.

For example: In the KGE model for the Beauty Dataset, as shown in **Figure 3.8,** embeddings of size 100 are created for each instance of the entities and relations.

```
model:  KnowledgeEmbedding(
        (user): Embedding(22364, 100, padding_idx=22363)
        (product): Embedding(12102, 100, padding_idx=12101)
        (word): Embedding(22565, 100, padding_idx=22564)
        (related_product): Embedding(164722, 100, padding_idx=164721)
        (brand): Embedding(2078, 100, padding_idx=2077)
        (category): Embedding(249, 100, padding_idx=248)
        (purchase_bias): Embedding(12102, 1, padding_idx=12101)
        (mentions_bias): Embedding(22565, 1, padding_idx=22564)
        (describe_as_bias): Embedding(22565, 1, padding_idx=22564)
        (produced_by_bias): Embedding(2078, 1, padding_idx=2077)
        (belongs_to_bias): Embedding(249, 1, padding_idx=248)
        (also_bought_bias): Embedding(164722, 1, padding_idx=164721)
        (also_viewed_bias): Embedding(164722, 1, padding_idx=164721)
        (bought_together_bias): Embedding(164722, 1, padding_idx=164721)
    )
```

Figure 3.8 KGE Model – Beauty Dataset

5. **Next Batch Processing**: Iterate from step 5 to step 6 while data remains.

6. **Load Batch Data**: Load the data from the dataset based on the batch size.

7. **Optimizer**: Use Stochastic Gradient Descent (SGD) optimizer to minimize the training loss and optimize the quality and efficiency of the embeddings.

8. **Checkpoint**: Save the checkpoint with their corresponding epochs.

9. **Model**: Load the last checkpoint from the final epoch.

10. **State Dictionary**: Retrieve the state dictionary from the model.

11. **Embeddings**: Extract the entities and relationships embeddings from the state dictionary obtained in the previous step.

12. **Save Embeddings**: Save the final embeddings.

13. **Refine Embeddings**: Refine the embeddings if necessary to improve their quality and performance.

14. **Finalize Embeddings**: Finalize the embeddings once they meet the desired criteria and are ready for use in further analysis or applications.

In summary, the objective of the KGE model is to minimize the smoothing loss resulting from the chosen parameters while developing robust embeddings of the KG structure. Incorporating KGE models like TransE is a strategic decision aimed at improving the usability of the KG. Converting the KG's complex network of entities and relationships into a compact numerical format enables efficient traversal and reasoning within the KG. This embedding process contributes to more accurate XR and insights, ultimately enhancing the value and utility of the KG.

c. KGE Model – Preserving Relationships and Semantics Captured within KG

Relationships and semantics captured within a KG are preserved and captured by KGE models through the representation of entities and relationships in a lower-dimensional space.

1. **Entity Embeddings**: KGE models map entities such as users, products, and categories to low-dimensional vectors. These embeddings capture the semantic meaning of entities based on their relationships with other entities in the KG.

   For example: **Figure 3.9**, depicts the embeddings of size 100 created for each instance of user and product entities.

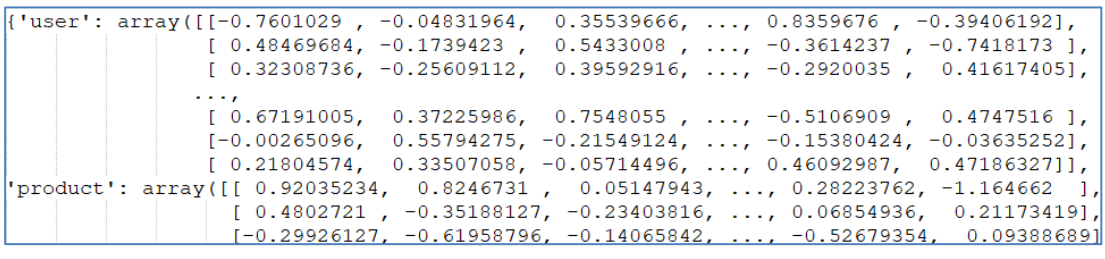

Figure 3.9 Embeddings of Size 100 of User and Product Entities

2. **Relationship Embeddings**: Relationships such as 'bought together' and 'belongs to' are also represented as low-dimensional vectors. These embeddings encode the semantic meaning of relationships by capturing the patterns of interactions between entities.

   For example: **Figure 3.10**, depicts a sample relationship vector with an embedding size of 100.

```
Create relation vector of size [1, embed_size].
- For all the relations (purchase, mentions, describe_as, produced_by,
                         belongs_to, also_bought, also_viewed, bought_together)
Ex:
    tensor([[ 4.9435e-03,  3.6995e-03, -4.1904e-03,  1.0566e-03,  1.9856e-03,
              1.4630e-03,  7.4911e-04, -2.4959e-04, -2.8742e-03, -4.4002e-03,
             -8.2954e-04, -6.0453e-04,  2.9000e-03,  4.2829e-03, -3.5156e-04,
             .....
             .....
              3.7957e-03, -5.5431e-04, -2.2367e-03, -1.3379e-03, -4.0801e-03]],
            requires_grad=True)
```

Figure 3.10 A Sample Relationship Vector with an Embedding Size of 100

3. **Training Process**: During training, KGE models learn to optimize the embeddings in a way that preserves the structure and semantics of the KG. This optimized embedding is achieved by minimizing a loss function that measures the discrepancy between observed and predicted relationships between entities.

4. **Semantic Proximity**: Entities and relationships that are semantically similar or related in the KG will have similar embeddings in the KGE model's embedding space. This similar embedding allows the model to capture complex semantic relationships.

5. **Efficient Representation**: By representing entities and relationships in a continuous, low-dimensional space, KGE models provide a more efficient representation of the KG. This low-dimensional representation enables faster computation and facilitates downstream tasks such as link prediction, recommendation, and semantic similarity analysis.

Overall, KGE model capture the relationships and semantics within a KG by transforming entities and relationships into compact, semantically meaningful embeddings that preserve the underlying structure and path patterns in the KG.

3.6.3 Develop Deep RL Actor-Critic Model

The primary objective of this phase is to construct a deep RL model capable of discerning user preferences and subsequently making personalized recommendations for content or products based on those preferences. However, identifying these user preferences is complex. To tackle this challenge, the approach undertaken in this study involves the utilization of deep RL.

The RL agent discerns patterns in user behaviour, particularly the paths they traverse, to enhance the rewards garnered. The RL agent refines its decision-making process by recognizing and learning from path patterns leading to favourable outcomes. The objective is to steer users along paths that yield greater rewards that are reflective of their authentic interests and preferences. Here, "reward" signifies the satisfaction or success linked with user interactions, such as completing a purchase.

a. Actor-Critic RL Model

This framework culminates in developing and deploying an Actor-critic RL model that takes advantage of the embedded KG as its operational environment. The model comprises two integral components: the Actor and the Critic.

- **Actor**: The Actor is tasked with decision-making within the KG environment, selecting actions based on its state and a defined policy.

- **Critic**: On the other hand, the Critic evaluates these actions and provides valuable feedback in the form of a reward signal.

This interaction between the Actor and the Critic constitutes a continuous learning process, where the Actor hones his decision-making skills based on the Critic's assessments. Throughout the training process, the Actor-Critic model is

guided towards acquiring the ability to adeptly navigate the intricate pathways within the KG. The training regimen focuses on empowering the Actor to understand the connectivity between users and products, enabling it to make informed decisions. By following relationships, uncovering interaction patterns, and leveraging the KG's structure, the Actor strives to optimize its actions to achieve favourable outcomes. With the support of the Critic's feedback loop, the Actor gradually refines its decision-making strategy, enhancing its proficiency in utilizing the information within the KG.

b. Deep RL Model

A Deep RL model is a type of ML algorithm that learns to make sequential decisions by interacting with an environment. It combines DL with RL principles to enable the model to learn complex behaviors and strategies autonomously (Heuillet et al., 2021). In the context of RS, a Deep RL model can be applied to learn optimal recommendation policies by directly interacting with users and their preferences.

This study employs deep RL, as shown in **Figure 3.11**, to learn actions undertaken by the RL agent within the KG environment, considering its current state and rewards received from actions in the previous KG state. This research adopts a DL model comprising four layers to implement the process of selecting optimal actions.

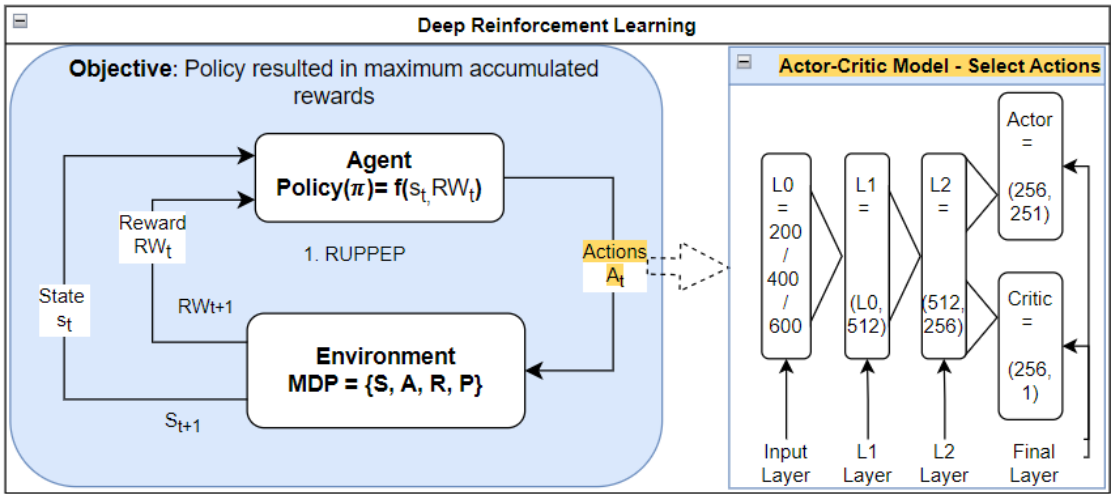

Figure 3.11 Deep RL Model – Used in this Research

The DL model has an input layer (L0) with state dimensions based on the hop count and the KGE model's embedding size: 200 for the first hop, 400 for the second, and 600 for the third. It includes two hidden layers of sizes 516 and 256. The output layer has an actor layer for action exploration and a critic layer for reward feedback.

c. RL Model – Path Patterns

In this context, "path patterns", as shown in **Figure 3.12** refers to the sequences of interactions that users make as they navigate the KG. By studying these patterns, the RL agent aims to uncover the most effective pathways that lead to favourable outcomes, such as purchases or interactions.

```
# Defining the followed path patterns in the KG
PATH_PATTERN = {
    # length = 3 - Final path
    1:  ((None, USER), (MENTION, WORD), (DESCRIBED_AS, PRODUCT)),
    # length = 3 - sub paths
    2:  ((None, USER), (MENTION, WORD), (MENTION, USER)),
    3:  ((None, USER), (PURCHASE, PRODUCT), (PURCHASE, USER)),
    4:  ((None, USER), (PURCHASE, PRODUCT), (DESCRIBED_AS, WORD)),
    5:  ((None, USER), (PURCHASE, PRODUCT), (PRODUCED_BY, BRAND)),
    6:  ((None, USER), (PURCHASE, PRODUCT), (BELONG_TO, CATEGORY)),
    7:  ((None, USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT)),
    8:  ((None, USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT)),
    9:  ((None, USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT)),
    # length = 4 - Final paths
    10: ((None, USER), (MENTION, WORD), (MENTION, USER), (PURCHASE, PRODUCT)),
    11: ((None, USER), (PURCHASE, PRODUCT), (PURCHASE, USER), (PURCHASE, PRODUCT)),
    12: ((None, USER), (PURCHASE, PRODUCT), (DESCRIBED_AS, WORD), (DESCRIBED_AS, PRODUCT)),
    13: ((None, USER), (PURCHASE, PRODUCT), (PRODUCED_BY, BRAND), (PRODUCED_BY, PRODUCT)),
    14: ((None, USER), (PURCHASE, PRODUCT), (BELONG_TO, CATEGORY), (BELONG_TO, PRODUCT)),
    15: ((None, USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (ALSO_BOUGHT, PRODUCT)),
    16: ((None, USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (ALSO_VIEWED, PRODUCT)),
    17: ((None, USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT)),
    18: ((None, USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (ALSO_BOUGHT, PRODUCT)),
    19: ((None, USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (ALSO_VIEWED, PRODUCT)),
    20: ((None, USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT)),
    21: ((None, USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (ALSO_BOUGHT, PRODUCT)),
    22: ((None, USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (ALSO_VIEWED, PRODUCT)),
    23: ((None, USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT)),
}
```

Figure 3.12 Path-Patterns for the RL Model

The research explores patterns of paths covering two or three hops, aiming for the final hop to converge on the PRODUCT node, with the USER node as the starting point. This exploration confines all possible relations and sub-nodes within the KG.

d. RL Model – Parameters

In the development of the RL model, parameters, as shown in **Table 3.6**, play a crucial role in creating a more robust RL model. The key parameters include "epochs", "batch-size", "learning-rate (lr)", "max_acts", "max_path_len", "act_dropout", "gamma", "ent_weight", and "steps_per_checkpoint".

Balancing these parameters ensures the RL model accurately reflects user behaviours, capturing paths to previously purchased products and preserving the semantic essence of entities and relationships. A larger batch size deteriorates model quality, while a smaller “steps_per_checkpoint” increases model generation time.

Table 3.6 RL Model -key Parameters

Parameters/Arguments	Descriptions	Default Value
dataset	One of {beauty, cd, cell, cloth}	BEAUTY
name	model name	train_RL_agent
epochs	max number of epochs to train	100
batch_size	batch size	32
lr	learning rate	1e-4
max_acts	Max number of actions	250
max_path_len	Max path length	3
gamma	reward discount factor	0.99
steps_per_checkpoint	Number of steps for checkpoint	5000
checkpoint_folder	Checkpoint folder name	checkpoint
is_resume_from_checkpoint	If resume the run from the last checkpoint state?	1
ent_weight	weight factor for entropy loss	1e-3
act_dropout	action dropout rate	0
hidden	number of samples	[512, 256]

e. RL Model – Algorithm

Below is the algorithm followed to develop the deep RL model for the research.

1. **Initialize Parameters**: Set the parameters as mentioned in section 3.6.3.d.

2. **KG Environment**: Generate KG Environment with KG State, getting Rewards, and possible Actions functionalities.

3. **Epochs**: Iterate through the following steps from step 4 to step 10 for the specified number of epochs.

4. **Train Model**: Train the Actor-Critic RL model.

   For example, **Figure 3.13,** depicts the Actor-Critic model with a state dimension (state_dim) of size 400 and an action dimension (act_dim) of size 251.

```
Ex: Parameters : env.state_dim:  400 env.act_dim:  251
    model : ActorCritic(
                        (l1): Linear(in_features=400, out_features=512, bias=True)
                        (l2): Linear(in_features=512, out_features=256, bias=True)
                        (actor): Linear(in_features=256, out_features=251, bias=True)
                        (critic): Linear(in_features=256, out_features=1, bias=True)
                      )
```

Figure 3.13 Actor-Critic Model Description

5. **Next Batch Processing**: Iterate from step 5 to step 9 while data remains.

6. **Load Batch Data**: Load the user data with their corresponding purchased labelled products from the dataset based on the batch size.

7. **Reached PRODUCT Node**: Iterate from step 8 to step 9 while data the hop doesn't terminate at the Product node.

8. **Load Batch State, Paths, Actions, and Rewards**: Load the batch state, paths, actions, and rewards from the KG environment.

9. **Optimizer**: Use Adaptive moment estimation (Adam) optimizer to minimize the training loss and optimize the quality and efficiency of the RL model.

10. **Checkpoint**: Save the checkpoint with their corresponding epochs.

Building upon the RL model, the research introduces the Guided Representation of User Preferences via the Path Embedding Propagation (RUPPEP) framework, which will be extensively discussed in the Chapter IV.

In summary, the objective of the RL model is to minimize the actor, critic, and entropy losses resulting from the chosen parameters while developing robust RL model. A key outcome of this phase is the development of a robust RL model. This model is designed to efficiently converge to optimal decision-making paths that lead to the recommendation of products that align with the user's preferences. By continually adapting and improving its recommendations based on observed user interactions, the RL model can enhance the accuracy and effectiveness of the product recommendations. A successful RL model contributes to a more refined recommendation process, guiding users toward products that resonate with their preferences and increasing the likelihood of positive interactions.

3.6.4 Recommendation Generation

Following the establishment of the RL model, the next step is recommendation generation, as shown in **Figure 3.5 (5).** This phase introduces various algorithms to streamline the recommendation process and generate explanations for the recommendations.

Recommendation generation poses a significant challenge. Among the many products available, selecting a few top products to recommend to users that resonate with their interests and prompt them to act is daunting. Additionally, incorporating explainability into the generated recommendations presents another major challenge. Considering the dual objectives of improving recommendations and generating corresponding explainabilities, this phase employs several specific algorithms, as detailed in the subsections below.

A more detailed discussions will be covered in the Chapter IV on this topic.

a. Max Explainability Score (MES)

There may be many possible traversal paths for reaching a product from the user node. The proposed algorithm, Max Explainability Score (MES), plays a crucial role in this fourth phase of the research by strategically identifying the connectivity path that produces the highest explainability for the user regarding a particular product.

b. Products Prioritisation Score (PPS)

Subsequently, the selected paths with MES scores yield a set of candidate products, each accompanied by the connectivity path from users that produces the highest explainability score. Given that all products have optimal paths from users that generate maximum explainability for those products, prioritization algorithms become essential to recommend products that align with users' aspirations.

The proposed algorithm, Products Prioritisation Score (PPS), assists with the product prioritization for the users. This algorithm engages to furnish users with selection of the most prioritized products.

c. Recommendation Generation and Explainability

Both MES and PPS assist the model to recommend recommendations to user that are having affinity to them.

After the recommendation phase, attention shifts to conveying the reasoning and rationale behind the product recommendations to the user. Here, the model implements a mechanism to extract insights and explanations from the trained RL model. By capitalizing on the connectivity path bearing the highest explainability score, the model crafts corresponding explanations for the recommended products. This approach ensures that users not only receive recommendations but also comprehend the underlying logic driving those suggestions.

Furthermore, the model introduces a quantitative metric to gauge the explainability of the provided recommendations. This metric termed the MES, offers a quantified evaluation of the transparency and comprehensibility of the recommended products. This holistic approach reinforces the interpretability and effectiveness of the generated recommendations.

d. How do Generated Explanations Help to Understand the Reasoning?

The study aims to enhance model explainability in KG-driven RSs by leveraging semantic information within the KG. Instead of providing post-hoc explanations, the approach equips the RL model to justify its predictions in real-time, drawing upon rich network relationships and contextual data. An embedded explainability mechanism rooted in KG principles is introduced, providing contextually relevant justifications for recommendations. This mechanism traces logical decision-making pathways within the KG, offering users transparent insights into recommendation rationale. Overall, the study integrates explainability seamlessly into the recommendation process, delivering tailored product recommendations supported by meaningful explanations derived from the KG's semantic structure.

3.7 PHASE 4: MODEL EXPERIMENTATION

The next phase involves conducting thorough experimentation with various settings and parameters. A model's journey remains incomplete until it undergoes thorough evaluation. The subsequent phase encompasses the critical stage of model evaluation, where the performance of the model is meticulously assessed across all four datasets. In pursuit of a comprehensive evaluation, prominent metrics such as NDCG, HR, Recall, and Precision stand as central tools for gauging effectiveness.

These key evaluation metrics collectively provide a robust framework to measure the model's efficacy in generating recommendations. NDCG encapsulates the ranking quality, HR quantifies the percentage of successful recommendations, Recall assesses the model's ability to capture relevant items, and Precision delineates the accuracy of the suggested products. Through the lens of these metrics, the model's

performance across diverse datasets is methodically analysed, offering insights into its strengths and potential areas of improvement.

A more detailed discussion of this topic will be covered in Chapter V.

3.7.1 Experimentation

In the experiment, different settings were employed to investigate various aspects of the model's performance. To ensure a robust evaluation, the experiment data was divided into separate training and test sets. The model was then developed using the training set, allowing it to learn patterns and relationships within the data. Subsequently, the model's performance was evaluated using the test set to assess its ability to generalize to unseen data. This process helps validate the model's effectiveness and provides insights into its performance under different conditions.

3.7.2 Model Evaluation

This phase assesses the effectiveness of the developed RL model using a set of well-defined evaluation parameters. These metrics gauge the model's performance in generating recommendations. Among the established evaluation parameters utilized are NDCG, Recall, HR, and Precision.

Besides these metrics, this research study proposes two more valuable evaluation metrics, MES and PPS, for identifying the best-suited traversal paths and corresponding affinity scores to the user for the candidate recommendations.

In conclusion, this phase is dedicated to evaluating the performance of the RL model's recommendations. It employs established metrics like NDCG, Recall, HR, and Precision to measure accuracy and relevance. Additionally, the study introduces a novel parameter, MES and PPS, which uniquely considers the model's embedded explainability, further contributing to the quality and transparency of the recommendations provided.

3.8 PHASE 5: ANALYSIS OF RESULTS

This phase delves into the baseline comparison and conducts the analysis and discussion of the results. A more detailed discussions will be covered in the Chapter VI on this topic.

3.8.1 Baseline Comparison

This study proposes a new XAIRec framework for XR with enhanced explainability. It is essential to assess the efficacy of the proposed framework by comparing its performance with different state-of-the-art baseline methods. This study compares the proposed model's effectiveness with various baseline models and other research work using the same dataset. This comparative analysis is a common and essential step in the research process to assess the effectiveness and advancement of the proposed approach. Here's a breakdown of the steps this study follows:

a. Reviewing with the Baseline Methods:

- The study started by reviewing the baseline model PGPR proposed by (Xian et al., 2019) in their research paper.

- This step involved understanding these baseline models' details, methodologies, and reported performance metrics.

b. Comparing the Proposed Model with the Baseline Methods:

- This study then compared the performance of the proposed model to the baseline model PGPR mentioned in (Xian et al., 2019) paper.

- This comparison likely involved evaluating various performance metrics, such as NDCG, precision, recall, and HR metrics.

c. Exploring other Research Works:

- This study expanded the analysis by exploring other ongoing research, ReMR (Wang et al., 2022), ECFKG (Ai et al., 2018), RKGR-RNS (Zhang et al., 2022), and HR-RLKG (Song et al., 2023), on the same dataset used in this study with the same objectives of XR.

- This step allowed us to consider the broader context of research in the field and identify how the model stacks up against the latest developments.

Overall, this rigorous comparison and evaluation process is crucial for validating the research contributions and highlighting any strengths or weaknesses of the model. It also helps the study position this work within the broader research landscape and demonstrate the uniqueness or improvements the model brings to the field. The outcomes of this comparison hold significant implications for the research. If the proposed model consistently outperforms or competes favorably with the baseline models across multiple effectiveness parameters, it substantiates the advancement and superiority of the novel approach. This empirical evidence highlights the viability and practicality of the proposed model in generating improved recommendations.

Furthermore, the comparison results offer insights into the strengths and weaknesses of the proposed model. Ultimately, highlighting the effectiveness of the proposed approach compared to state-of-the-art baseline models strengthens the research's contribution to the field. It demonstrates that the novel ideas, methodologies, and mechanisms incorporated into the proposed model translate into tangible improvements in recommendation quality and their associated explainability. This empirical validation builds credibility for the research and highlights its potential impact in real-world applications where accurate and transparent recommendations are crucial.

3.8.2 Results Analysis

The ensuing step involves the analysis of these results. Beyond mere data, this phase delves into the implications derived from the outcomes. The model's explainability aspect is also given due attention. An illustrative depiction of model explainability is provided, encompassing the process of generating these explanations and their subsequent evaluation. This serves to demystify how the model's recommendations are rationalized and presented to users.

Holistically, the findings are then synthesized with the research objectives. Their alignment is scrutinized, leading to a comprehensive discussion that delves into the implications of the results. This discussion bridges the gap between the study's aims and the outcomes achieved, elucidating how the model's performance, explainability, and overall impact intertwine with the original research objectives.

3.9 PHASE 6: CONCLUSION

The culminating phase encompasses the conclusion and future work, offering a comprehensive wrap-up of the research journey. The conclusion stage distils the research endeavour into a succinct summary, encapsulating the key findings, contributions, and overarching significance. It presents a cohesive synthesis of the research's core outcomes and how they align with the original objectives.

Moreover, the future work segment bridges the current study and the evolving landscape of research. It recognizes any limitations inherent to the study and underscores the potential avenues for future research. These proposed directions aim to refine and advance the existing work, paving the way toward achieving the broader vision of XAI.

In essence, the conclusion provides closure by encapsulating the study's achievements, while the future work section opens doors to uncharted territories, signalling an ongoing commitment to advancement and progression in the realm of XAI.

Overall, this research strategy is comprehensive and well-structured and involves a systematic progression from identifying the problem to proposing a solution and validating its effectiveness through experimentation. This approach helps ensure that this study contributes meaningfully to the field and provides valuable insights for future research and applications. A more detailed discussion of this topic will be covered in Chapter VII.

The operating model's concluding phase involves synthesising the entire research into a comprehensive and structured thesis report. This report is organized into distinct chapters, each serving a specific purpose and contributing to the overall narrative of the research findings. This phase aims to present the research, its methodologies, results, and implications coherently and logically, thereby communicating the significance of the work conducted.

3.10 SUMMARY

This chapter discusses the detailed methodology of this research. The chapter presented comprehensive procedures to solve the research problems and address the research questions. It describes the general flow of the research and the detailed operational framework. The operational framework involved distinct phases in achieving the research objectives. These phases range from the literature review to the final report writing. The research focused on enhancing RSs through a combination of KG utilization and RL techniques using the Amazon e-commerce datasets. The primary aim was to develop a robust model that provides accurate recommendations and offers embedded explainability for its decisions.

The research commenced with data preprocessing, involving extracting and structuring information from Amazon e-commerce datasets. The establishment of a KG was pivotal, as it served as a structured foundation for subsequent steps. KG embedding techniques, notably the Translational Algorithm (TransE), were applied to map entities and relationships into a lower-dimensional space. This facilitated efficient traversing of the complex graph structure. The RL model was the focal point of the research, driven by the objective of understanding user preferences and providing tailored recommendations. Path patterns reflecting typical user interactions

were identified through RL. The RL agent learned to navigate these paths, enhancing its rewards by recommending products that align with user preferences. The chapter also discusses the steps to generate recommendations using MES and PPS.

The model's efficacy is evaluated against state-of-the-art baseline models, employing metrics like NDCG, Recall, HR, and Precision.

Finally, the chapter discusses the culmination step of any research, i.e., developing a structured thesis report. This report synthesized the research process, methodologies, results, and implications across distinct chapters. Ultimately, the study contributed to advancing RSs by showcasing the effectiveness of KG-embedded RL models with real-time explainability.

CHAPTER IV

XAIREC: PROPOSED FRAMEWORK

4.1 INTRODUCTION

This chapter delves into a comprehensive examination of a proposed framework that addresses the challenges surrounding transparency and interpretability in RSs. It formalizes the concept of XR and introduces a framework named XAIRec, which is designed to tackle the hurdles related to transparency and interpretability.

The chapter initiates by underlining the significance of problem formulation and delineating the essential elements that drive the functionality of the framework. It subsequently delves into each element methodically.

A significant portion of the chapter delves into the algorithms that enhance explainability, introducing the RUPPEP component for learning user preferences and the MES algorithm for quantifying explainability. These approaches aim to build user trust by providing transparent reasoning behind recommendations. The chapter also introduces the PPS algorithm, which identifies suitable products and improves recommendation accuracy, reinforcing user trust in RSs.

In summary, this chapter serves as a guide, starting with the conceptualization of XR through the XAIRec framework and progressing to detailed algorithmic strategies. It highlights the importance of transparency and advanced algorithms in helping users make optimal choices from a range of options. The chapter is designed to cover the following sections and proceeds as follows: Section 4.2 elaborates on the detailed methodologies employed, encompassing research problem formulation and comprehensive descriptions of key components. Section 4.3 expands the training

phase of the framework by detailing the RUPPEP algorithm, while section 4.4 focuses on the recommendation phase by detailing the MES and PPS algorithms, explaining their roles within the framework, and discussing a quantitative explainability metric for evaluating recommendations. Section 4.5 provides a summary of the chapter.

4.2 XAIREC - FRAMEWORK

This section formalizes the XR problem and proposes an advanced approach named XAIRec to address the challenges of transparency and interpretability.

4.2.1 Preliminaries and XR Problem Formalization

Based on the work of (Xian et al., 2019), a KG $\mathcal{G}know$ is a directed graph.

$$\mathcal{G}know = \{(e_h, r, e_t) | e_h, e_t \in \mathcal{E}, r \in \mathcal{R})\} \quad \ldots(4.1)$$

It contains entity set $\mathcal{E}$ and relation set $\mathcal{R}$. Each edge represented as a triplet of the form (e_h, r, e_t) represents a fact of the relation r from the head entity e_h to the tail entity e_t. This study considers a specific type of KG for XR, denoted by $\mathcal{G}_{\mathcal{XR}}$. It contains a subset of User entities $\mathcal{U}$, and a subset of Item entities $\mathcal{V}$, where $\mathcal{U}, \mathcal{V} \subseteq \mathcal{E}$, and $\mathcal{U} \cap \mathcal{V} = \emptyset$. These entities are connected via relations r_{ui}, where $r_{ui} \in \mathcal{R}$.

a. *Definition* 1: ($k-hop$ path)

A k-hop path from entity e_0 to entity e_k is defined as a sequence of $k+1$ entities connected by k relations, denoted by

$$\mathcal{P}_{e_0 e_k}(e_0, e_k) = \{e_0 \overset{r_{e_0 e_1}}{\longleftrightarrow} e_1 \overset{r_{e_1 e_2}}{\longleftrightarrow} \cdots \overset{r_{e_{k-1} e_k}}{\longleftrightarrow} e_k\} \quad \ldots(4.2)$$

where $(e_{k-1} \overset{r_{e_{k-1} e_k}}{\longleftrightarrow} e_k)$ represents either $\left(e_{k-1}, r_{e_{k-1} e_k}, e_k\right) \in \mathcal{G}_{\mathcal{XR}}$ or $\left(e_k, r_{e_k e_{k-1}}, e_{k-1}\right) \in \mathcal{G}_{\mathcal{XR}}$.

b. *Definition* 2: (XAIRec-Problem)

Given a KG $\mathcal{G}_{\mathcal{XR}}$, user $u \in \mathcal{U}$, item $i \in \mathcal{V}$, and integers *K* and *N*, the goal is to find a recommendation set items $\{i_n\}_{n\in[N]} \subseteq \mathcal{V}$, such that each pair (u, i_n) is associated with an optimal (o) reasoning path.

$$\mathcal{P}_k^o\ (u, i_n)\ (2 \leq k \leq K) \qquad \ldots(4.3)$$

N is the number of recommendations that drives the recommendation of that item for the user and provides explainability for the recommendation. $\mathcal{P}_k^o\ (u, i_n)$ provides the optimal path set of the user u to the recommended item set i_n through a *k*-hop path.

The primary challenge revolves around recommending a set of products, denoted as $\{i_n\}$, which achieves dual objectives: firstly, it offers comprehensible explanations for the recommended products, and secondly, it improves the overall efficacy of the recommendations. Additionally, an issue arises concerning the assessment of the explanatory effectiveness, typically evaluated through qualitative methods due to the absence of robust quantitative measurement mechanisms.

4.2.2 XAIRec Framework

For providing the ideal item recommendations $\{i_n\}_{n\in[N]}$ and explainability, i.e., reasoning $\mathcal{P}_k^o\ (u, i_n)$ associated with those recommendations, the recommendation model needs to identify the optimal path to the recommended items for the user in the KG, providing explainability for the generated recommendations. Additionally, it should prioritize the recommended products through the affinity of them for the users.

Figure 4.1 represents a visual of the functioning of the detail-level design and methodology of the XAIRec framework, where it demonstrates the functioning of the model training and the recommendation phases by applying advanced algorithms of RUPPEP, MES, and PPS.

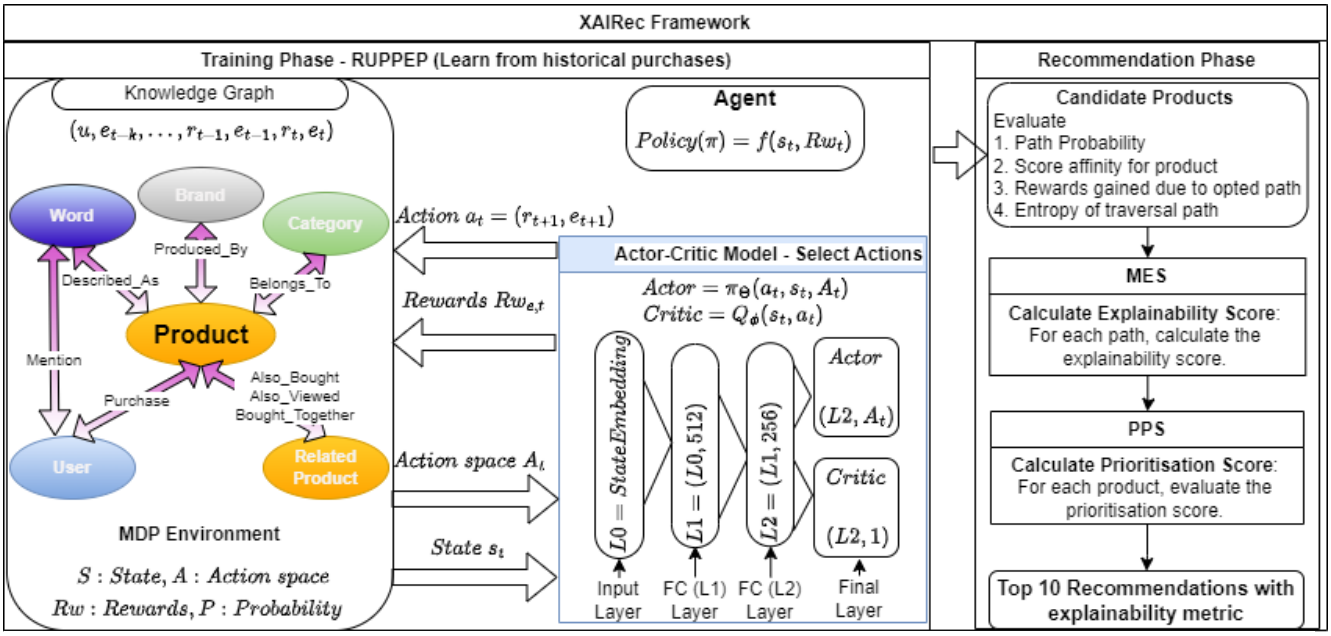

Figure 4.1 XAIRec - Framework Design and Architecture

The XAIRec framework comprises two main phases: training and recommendation. During the training phase, the framework employs the RUPPEP component, a deep RL-based component tasked with capturing user preferences by training the RL agent model to identify preferable path patterns users may follow. This component establishes the MDP environment using the KG and KGE developed earlier, as detailed in Chapter 3. The RL agent interacts with the MDP environment, receiving state and action spaces, and invokes the Actor-Critic model to determine probable actions. Based on the chosen action, the agent may receive positive or negative rewards and adjust its behaviour accordingly in subsequent iterations. The RUPPEP component iterates in batches for all users in the dataset, learning the best policy based on users' previously purchased products.

During the recommendation phase, the framework assesses the key parameters outlined in **Figure 4.1** and calculates the explainability score for each path from the user node to the candidate products. The MES algorithm identifies the path with the maximum explainability score. The next step involves recommending products based on the user's affinity for them. These phases will be elaborated upon in the below sections.

4.3 XAIREC – TRAINING PHASE

This section delves into the details of the training phase of the XAIRec framework. The central component of this phase is RUPPEP, which aids the framework in capturing user preferences. The specifics of RUPPEP and its role in the training phase will be discussed in detail here.

4.3.1 RUPPEP – Maximize RL Rewards and Path Preferences

The RUPPEP component trains the Actor-Critic model by utilizing historical purchasing information in a batched manner. The goal is to learn path preferences and generate enticing recommendations and tailored explainability. The RUPPEP component operates by leveraging the RL environment to motivate the agent's actions through a reward mechanism. This motivation is rooted in encouraging the agent to follow specific path patterns that lead to products the user has previously purchased. These purchased products, referred to as "Train Labels," are visibly marked in the diagram. Historical purchase information enables the framework to train the model, refining its recommendations by learning from and adapting to user preferences.

Figure 4.2 illustrates the working mechanism of the RUPPEP component. Consider a user named Adam, who has previously purchased Product B, Product D, and Product E. The KG contains multiple paths that Adam might have followed to reach these products, representing different decision patterns that contribute to understanding his preferences and behaviour.

The diagram shows limited scenarios using relationships such as PURCHASE, DESCRIBE, and MENTION. In the MDP environment, the agent can choose a path leading to a PURCHASE action or a MENTION action for Adam. The PURCHASE action indicates that Adam previously bought "Product A", while the MENTION action suggests a word mentioned by Adam.

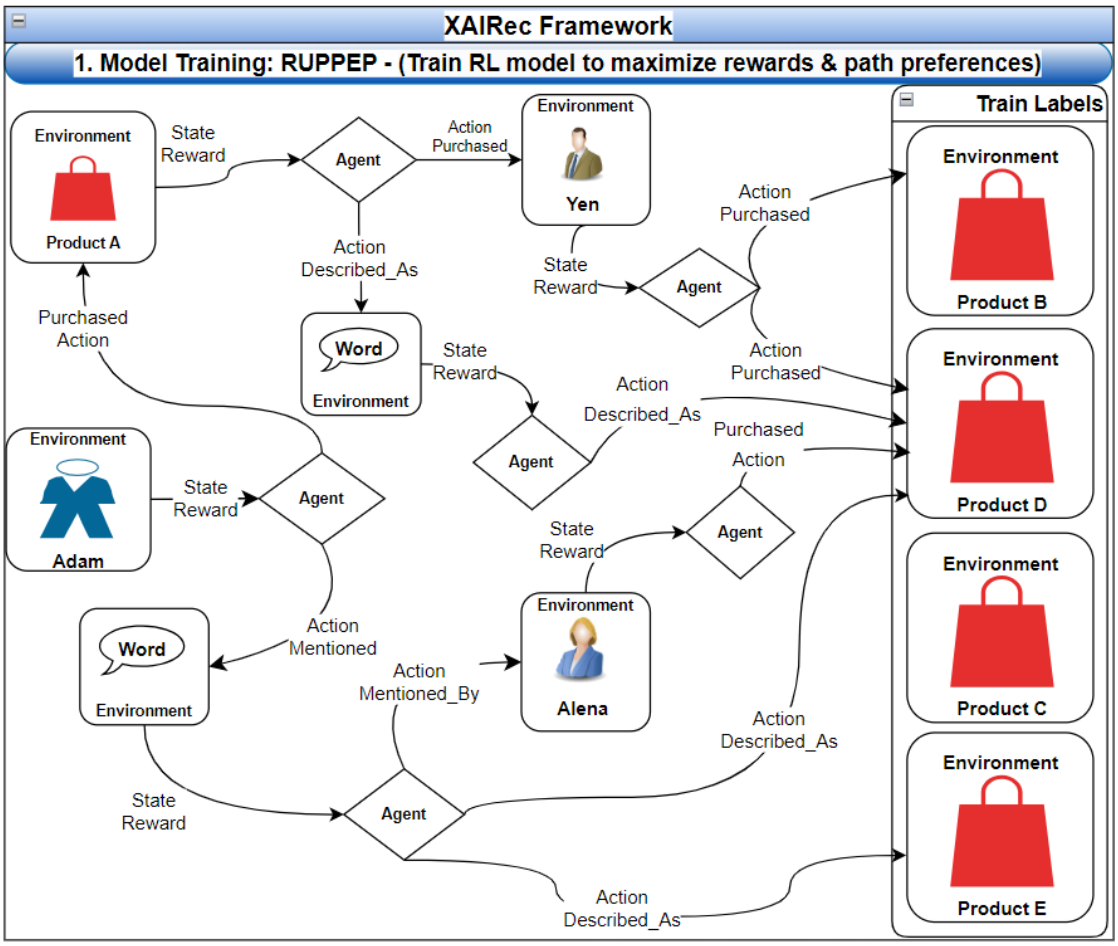

Figure 4.2 RUPPEP – an Illustration

Based on probabilities and path-traversal algorithms designed under the RUPPEP framework, the agent selects the most suitable path, leading to decisions influenced by received rewards and new probable actions from the intermediary node. For example, suppose the agent chooses the "Product A" path. In that case, it may need to decide further based on available actions, such as whether another user named Yen also purchased Product A or if certain words describe Product A.

The RUPPEP component recognizes that there might be multiple ways for Adam to reach the same products, and some of these paths might have enticed the RL agent to maximize rewards. Essentially, when Adam chose a particular path that led to a product, the RL agent received positive rewards for successfully guiding Adam. The RL agent learns from these patterns, internalizing the relationships between products and the routes Adam has historically favoured. By doing so, the agent effectively

deciphers the path affinities that Adam typically follows to discover products he might have a predilection for and get an awareness of user preferences.

a. Algorithm Functioning

The RUPPEP component aims to refine RSs through historical purchase information within a KG. The core element of the RUPPEP component consists of an improved reward function, an efficient path-finding algorithm, and the incorporation of edge-pruning techniques applied within the RL framework during the RS model training.

i. Reward Function ($\mathbb{R}w$)

The agent under an environment state takes some action from the available actions pool. The agent's selected action leads the current environment state to a new environment state. The reward function controls the agent's learning behaviour, either getting a reward or punishment for the selected action (Heuillet et al., 2021). This process iterates till the agent reaches the optimal state. The reward function encourages the system to adhere to path patterns that maximize the generated rewards and lead to products previously purchased by users. This serves to reinforce behaviour aligning with user preferences. The reward function is defined in Eq. 4.11.

ii. Edge-Pruning Techniques

Edge pruning is an integral aspect of the component, addressing uncertainties related to the relevance of specific items to users' preferences. Given the diverse and heterogeneous nature of information in the KG, certain connections may carry more uncertainty, and edge pruning mitigates this issue.

This study evaluates the matrix multiplication scores of [USERS] embeddings by combining them with the corresponding relationship embeddings and all possible candidate actions. It then selects the allowable actions based on the scores in descending order and prunes the rest. The idea is to prune edges that are less important for users, ultimately reducing the action space in the RL environment. The code of the edge pruning is depicted in **Figure 4.3**.

```
# (5) If there are too many actions, do some deterministic trimming here!
user_embed = self.embeds[USER][path[0][-1]]
scores = []
for r, next_node_id in candidate_acts:
    next_node_type = KG_RELATION[curr_node_type][r]
    if next_node_type == USER:
        src_embed = user_embed
    elif next_node_type == PRODUCT:
        src_embed = user_embed + self.embeds[PURCHASE][0]
    elif next_node_type == WORD:
        src_embed = user_embed + self.embeds[MENTION][0]
    else:  # BRAND, CATEGORY, RELATED_PRODUCT
        src_embed = user_embed + self.embeds[PURCHASE][0] + self.embeds[r][0]
    score = np.matmul(src_embed, self.embeds[next_node_type][next_node_id])
    # This trimming may filter out target products!
    # Manually set the score of target products a very large number.
    if is_train == 1 and next_node_type == PRODUCT and next_node_id == target_product:
        score += 99999.0
    scores.append(score)
candidate_idxs = np.argsort(scores)[-self.max_acts:]  # choose actions with larger scores
candidate_acts = [candidate_acts[i] for i in candidate_idxs]  # New code block
actions.extend(candidate_acts)
return actions
```

Figure 4.3 Code Snippet – Edge Pruning

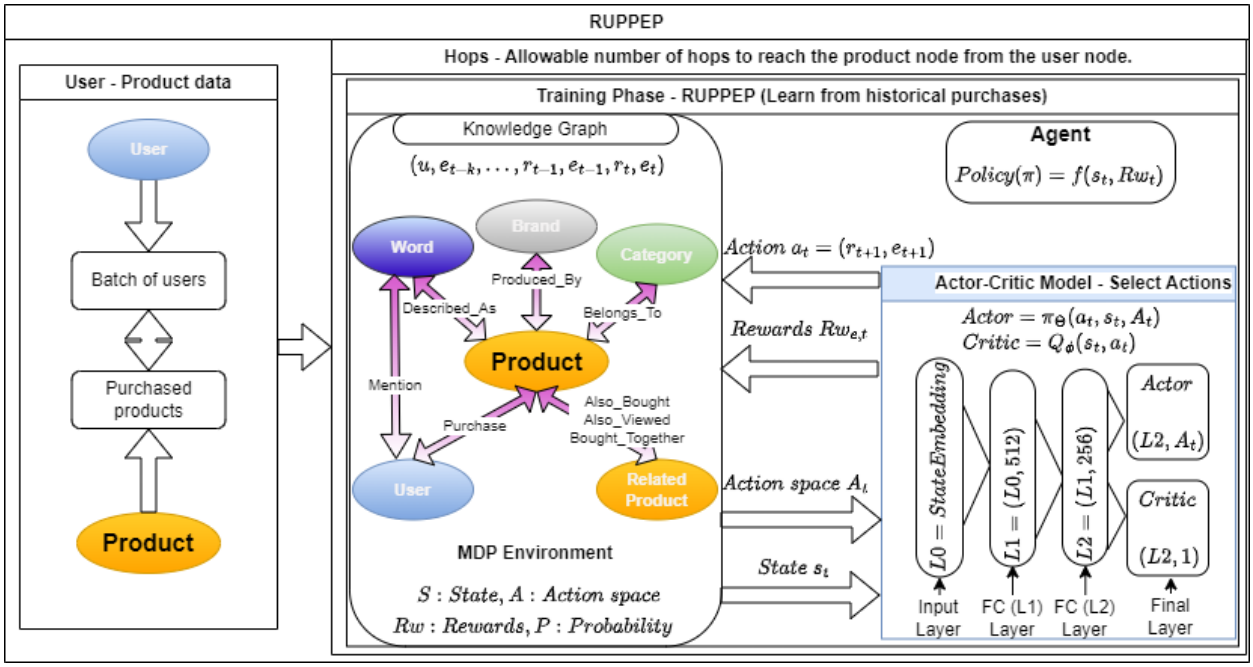

Figure 4.4 RUPPEP – The Pathfinding Algorithm - Design

4.3.2 Pathfinding Algorithm: RUPPEP – Train $\boldsymbol{ActorCritic}$ Model

As depicted in **Figure 4.4**, the pathfinding algorithm navigates the KG effectively and aids in identifying relevant connections between products and user preferences. The main idea is to train an RL agent in a KG environment to learn effective policy (π).

The agent learns to navigate towards potentially "good" items, particularly those previously purchased by the user. By understanding the path patterns, the agent adjusts parameter weights at each hop in the traversal paths. Enumerating all possible traversal paths between users and potential item entities in large graphs can be computationally expensive. Therefore, graph pruning techniques using rewards as a heuristic function reduce the search space and improve efficiency in finding relevant paths. The objective is to achieve maximum reward by efficiently sampling reasoning paths for each user, leading to the previously purchased items.

a. Algorithm - RUPPEP

Algorithm 1: *RUPPEP - The Guided Representation of User Preferences via the Path Embedding Propagation*

INPUT: A set of Users $\boldsymbol{\mathcal{U}}$, A set of Items $\boldsymbol{\mathcal{V}}$, KG $\boldsymbol{\mathcal{G}_{\mathcal{XR}}}$, Batch Processing size $\boldsymbol{b}$

OUTPUT: A better trained RL $\boldsymbol{ActorCritic}$ model

// Develop an MDP environment

1 *Develop an MDP environment using the KG and KGE as the foundation.*
2 *Initialize the KG state based on the size of the embedding vector and the current hop.*
3 *Develop a function to calculate MDP rewards for batch processing.*
4 *Develop a function to identify the MDP action space for batch processing.*
5 *Develop a function to generate the current MDP state for batch processing.*

// Initiate the Actor-Critic model

6 Initiate the input layer (L0) of the Actor-Critic model with the State dimension

7 Add two hidden layers with a size of 512, and 256.

8 Define the Actor layer with the allowable action dimensions

9 Define the Critic layer with the corresponding feedback

10 Initiate the Adam optimizer to optimize the model parameters.

// Core processing logic of the RUPPEP component

11 Iterate for the number of allowable epochs

// Identify the batch users and their prior purchased products for developing the Actor-Critic RL model

12 {u} ← select *($\mathcal{U}$, b)▷ A set of users from $\mathcal{U}$ of a batch size (b)*

13 $\{v\} \leftarrow$ *get_product ($\mathcal{V}$, {u}, 'Purchase') ▷ set of purchased products for batch users {u} from the train dataset.*

// Train the agent to adapt to user purchase behaviour. The agent will get maximum rewards if the path follows the set pattern and ends to the i'th targeted products $\{v\}_i$.

14 for i in range ($len(\{v\})$:

a. Reset the State, Actions, and Rewards for the set of users {u} *to set of i'th products* $\{v\}_i$

b. For hop in range (3):

*i. Agent **selects** the actions out of probable actions in the $\mathcal{G}_{\mathcal{XR}}$.*

*ii. Environment provides **the next set of actions**, new state, path followed, and the rewards based on the current state and actions.*

iii. The agent minimizes the Actor-Critic model loss based on the rewards received from the MDP environment and adjusts the model parameters accordingly using the Adam optimizer.

1. The optimizer function is invoking the below key functions

*a. **Optimizer.zero_grad**(): It makes the gradients of all*

the model parameters to zero.

b. ***Loss.backward****(): It computes gradients of loss with respect to all the parameters in the loss function and updates the gradients.*

c. ***Optimizer.step****(): It updates all the model parameters based on the gradients.*

// Save the checkpoint of the RL model

15 *Save the learned policy of the Actor-Critic model to a checkpoint file at the end of each corresponding epoch.*

b. Deciphering the Algorithm

The following steps are elaborated on as stated in Algorithm 1.

1. Develop an MDP environment.

2. Develop an Actor-Critic model.

3. Select a set of users {u} of a batch size b. At the outset, a group of users is chosen based on a predetermined batch size.

4. Get the set of purchased products $\{v\}$ for those users from the training dataset. This data forms the foundation for the subsequent RL process, as it constitutes the user-product interactions on which the RL agent will make decisions.

5. Train the RL agent to follow the connectivity path patterns with a maximum of 3 hops. The core of the process involves training a RL agent. The RL agent learns through trial and error to navigate a network of connectivity path patterns to reach the purchased products. These path patterns consist of

interconnected nodes, with a maximum of 3 hops (intermediate steps) allowed. The goal of the RL agent is to make decisions (actions) that lead to the target purchased products, optimizing rewards in the process.

6. During the process, the RL agent optimizes the overall rewards by selecting an action out of many probable actions in each state. The RL agent's decision-making is driven by maximizing cumulative rewards. At each step, the agent evaluates the potential actions it can take, aiming to select the action that is most likely to lead to the target purchased product and consequently result in the highest reward.

7. RL Agent receives intermediary sub-path rewards positively or negatively. As the RL agent progresses along the connectivity path pattern, it receives intermediary rewards. These rewards are either positive or negative, depending on whether the path the agent is following adheres to the predefined set path pattern and ultimately guides the agent towards the target purchased product. Positive rewards reinforce favourable behaviour, while negative rewards discourage undesirable actions.

8. The Actor-Critic model iterates the process for all the users in multiple epochs. The Actor-Critic model is a type of RL architecture that combines the strengths of both policy-based (Actor) and value-based (Critic) methods. This model iterates over the selected batch of users' multiple times (epochs) to refine the RL agent's decision-making. Over time, the agent learns to make more informed decisions based on the rewards it receives and the paths it takes.

This entire process encapsulates a proactive approach to understanding user preferences by training an RL agent to navigate through connectivity path patterns in an RL environment designed by utilizing the constructed KGE. By optimizing rewards and iteratively learning from interactions with the training dataset, the Actor-Critic model helps uncover underlying user preferences and behaviour patterns.

4.4 XAIREC – RECOMMENDATION PHASE

This section delves into the details of the recommendation phase of the XAIRec framework. The central components of this phase are MES and PPS, which aid the framework in recommending prioritized products to users based on their affinities for products, considering both user preferences and model-intrinsic explainabilities. The specifics of MES and PPS and their roles in the recommendation phase will be discussed in detail here.

4.4.1 Recommendation Phase – Building Blocks

The following are the building blocks and core concepts used in this section.

- $\mathbb{R}$ - the number of hops of the traversal path.
- $\mathbb{P}$ - the probability of the traversal path.
- $\mathbb{H}$ - the entropy/information value due to the opted traversal path.
- $\mathbb{R}w$ - the reward of the traversal path.
- $\mathbb{S}$ - the affinity score of the product for the user.

a. Rules ($\mathbb{R}$)

$\mathbb{R}$ defines the number of rules considered while determining the recommendations' explainability. The rules here may comprise the number of features required to generate the explainability. In a KG, the rules may comprise the number of hops required to generate the explainability. This measure is predicated on the notion that more straightforward explanations are preferable, based on the work of (Gigerenzer & Brighton, 2009) on bias-variance trade-offs. Thus, $\mathbb{R}$ is defined as follows, where |Nodes| is the number of nodes/features required to generate the explainability.

$$\mathbb{R} = \boldsymbol{Count(nodes)\ OR\ Count(Hops) = |nodes|} \quad \ldots(4.4)$$

b. Path Probability (ℙ)

ℙ represents the likelihood that the user will be interested in the recommended products, considering the path they followed to discover them. As per the meta-graph, many traversal paths could be from the user node to the product nodes with different multiplicative probabilities. Products with a better path probability might have a better affinity of users towards liking the same. The path probability is determined by the logits and softmax functions along the path from the user node to the product nodes, as defined in Equation 4.6.

i. Logits

Figure 4.5 depicts the evaluation of the logits for the Actor model. This deep layer employs dropout functionality, which randomly drops nodes with a specified dropout probability. The activation function used in this model is the exponential linear unit (elu).

```
state, act_mask = inputs  # state: [bs, state_dim], act_mask: [bs, act_dim]
x = self.l1(state)
x = F.dropout(F.elu(x), p=0.5)
x = self.l2(x)
x = F.dropout(F.elu(x), p=0.5)
actor_logits = self.actor(x)
```

Figure 4.5 Code Snippet – Evaluation of Actor Logits Function

ii. Softmax

After evaluating the logits for the Actor model, the next step is to apply the softmax function to convert the logits into their corresponding probability values. The logits are vectors, and the softmax function allows these vectors to be converted into probability values (Li et al., 2020).

iii. Path

A $Path$ may contain many nodes, and the multiplicative value of their corresponding node probabilities provides the probability of following a particular path.

iv. Probability

In deep RL, the probability of the agent hopping from the current node to a node connected to the current node is defined as based on the current state of the environment and the available actions, as shown in the following equation, where P($node_i$) is the probability of a node i in each path.

$$\boldsymbol{P(node) = softmax\big(logits(State, Action)\big)} \quad \ldots(4.5)$$

Based on the node's transition probabilities, the path probability can be evaluated by multiplying the transition probabilities of nodes in a path.

$$\mathbb{P} = \boldsymbol{Path\big(P(node)\big) = \prod_{i=1}^{n} P(node_i)} \quad \ldots(4.6)$$

c. Entropy ($\mathbb{H}$)

The concept of surprise in KGs is related to the idea of unexpected or surprising connections between entities in the graph. Since explainability relates to the surprise factor or interest of the user for the recommended product, the higher surprise factor may relate to higher user satisfaction with the presented explanation. The research of (Klein et al., 2022) examining surprise facts in Wikidata lends credence to this metric as it reveals that an entity is judged to be more unexpected for a fact if it is distinct from entities that have the fact. Similarly, (Tsurel et al., 2017), in their search for surprising publications, propose that an item is considered surprising if it deviates significantly from the norm for its category.

Explanations with higher information values have better explainability than those with lower information values. The information value of a content, statement, or fact depends upon the entropy of that information. The entropy relates to the surprise

components produced by the content. It provides a way to quantify the amount of surprise in some information (in this case, the explanation). The KG framework may have many paths from one node or entity to another targeted node or entity. The entropy of these traversed paths helps to get the information values generated by the statements comprising the high surprise components to provide good, reasonable explanations to the user for every recommendation. Thus, $\mathbb{H}$ is defined as the entropy of the path as follows.

$$\mathbb{H} = \boldsymbol{Information\ Value(Path)} = \boldsymbol{Entropy(Path)} = -\sum_{i=1}^{n} \boldsymbol{P(node_i) * log\ (P(node_i))} \quad \ldots(4.7)$$

Here, the $log(P(node_i)$ provides the logarithmic value of the probability of the i'th node in the path traversed in the KG. The entropy needs to be normalised to balance the number of features used for generating the explanations, as follows.

$$\boldsymbol{Norm(Entropy(Path))} = \frac{-\sum_{i=0}^{n} \boldsymbol{P(node_i) * log\big(P(node_i)\big)}}{\boldsymbol{|nodes|}} \quad \ldots(4.8)$$

d. Affinity Score ($\mathbb{S}$)

$\mathbb{S}$ defines the affinity score of the product recommendations for the user based on his past purchases and liking or inclination towards other similar products. Users with historical purchases of products would have some affinity towards other similar products. The score here defines the cosine similarity of the user's previously purchased products with other products and provides a user-like score for those products (Lawrence et al., 2001).

The KGE assists in evaluating the recommendation score of the user for the product in the KG environment, as follows.

$$\boldsymbol{User_{Embed} = Embed(User)}$$
$$\boldsymbol{Purchase_{Embed} = Embed(Purchase)} \quad \ldots(4.9)$$

$$Product_{Embed} = Embed(Product)$$

The $User_{Embed}$, $Purchase_{Embed}$, and $Product_{Embed}$, as depicted in (eq. 4.9) refer to the embedding vectors of the User, Purchase, and Product nodes in the KG, each with an embedding size of 100. The (+) operator, sums the $User_{Embed}$ with $Purchase_{Embed}$, as depicted in (eq. 4.10). The numeric vector of $Purchase_{Embed}$, is added to each individual vector of $User_{Embed}$.

$$UserPurchase_{Embed} = User_{Embed} + Purchase_{Embed} \quad \ldots(4.10)$$

The score of a user's affinity for purchasing the product can be defined as the cosine similarity of the $UserPurchase_{Embed}$ and $Product_{\text{Embed}}$, calculated through the DOT ($\odot$) operator as depicted in Eq. 4.11. This cosine similarity provides the affinity score for every product in the $Product_{\text{Embed}}$ with respect to every user vector in the $UserPurchase_{Embed}$.

$$\mathbb{S} = \odot(UserPurchase_{Embed}, Product_{Embed}) \quad \ldots(4.11)$$

e. Rewards ($\mathbb{R}w$)

The reward function leads to the recommendation score and utilizes the $PATH\_PATTERN$, prescribed in the section 3.6.3.c. The higher reward function value leads to an optimal path for that recommended product. Eq. 4.8 and Eq. 4.9 assist in evaluating the reward function score of a user's affinity for purchasing the product, and $\mathbb{R}w$ is defined as in Eq. 4.12:

$$\mathbb{R}w = \begin{cases} 0\ ; if\ len\ (path) \leq 2 \\ +ve\ Rewards\ ;\ if\ len(path)\ \geq 3\ and\ follows\ PATH\_PATTERN \\ High + ve\ Rewards\ ; if\ len(path) \geq 3\ and\ follows\ PATH\_PATTERN\ and\ node\ type\ is\ "Product" \\ -ve\ Rewards\ ;\ if\ len(path)\ \geq 3\ and\ NOT\ follows\ PATH\_PATTERN \end{cases} \quad \ldots(4.12)$$

The reward function returns zero if the path length does not meet the requirement of the minimum number of hops that the user should have traversed. The agent gets rewarded for following the prescribed $PATH\_PATTERN$ described by the environment or else punished. The agent receives a maximum reward if the final node type is the destined node type, i.e., the "$Product$".

4.4.2 Model Recommendation: MES - Efficient Pathfinding

Once a robust recommendation model has been developed using the training dataset, the subsequent crucial phase involves utilizing this model to generate product recommendations that not only resonate with user interests but also possess the capability to offer meaningful explanations. This objective entails enabling the model to be aware of efficient traversal paths within the underlying data structure, facilitating a clear rationale for the suggested recommendations.

The importance of these efficient and diverse traversal paths lies in their potential to furnish well-resonated explanations. These explanations serve as a means for the model to provide insights into why a particular recommendation is being made, thus enhancing the transparency and interpretability of the recommendation process.

a. Max Explainability Score - Definition

This study introduces an advanced algorithm, MES (Eq. 4.15), for quantifying the explainability of different paths from the user's perspective to the recommended product. It determines the optimal explainability path among the available options. In other words, it identifies the traversal path that provides the most comprehensive and coherent reasoning for the user to understand and accept the recommendation.

In summary, this explainability score encompasses two key aspects: First, it seeks to enhance the user's experience by maximizing the rewards linked to the path. Second, it strives to introduce novelty and diversity into the efficient path by maximizing the entropy associated with the traversal route. Consequently, this algorithm selects the path with the highest explainability score among various explainability options.

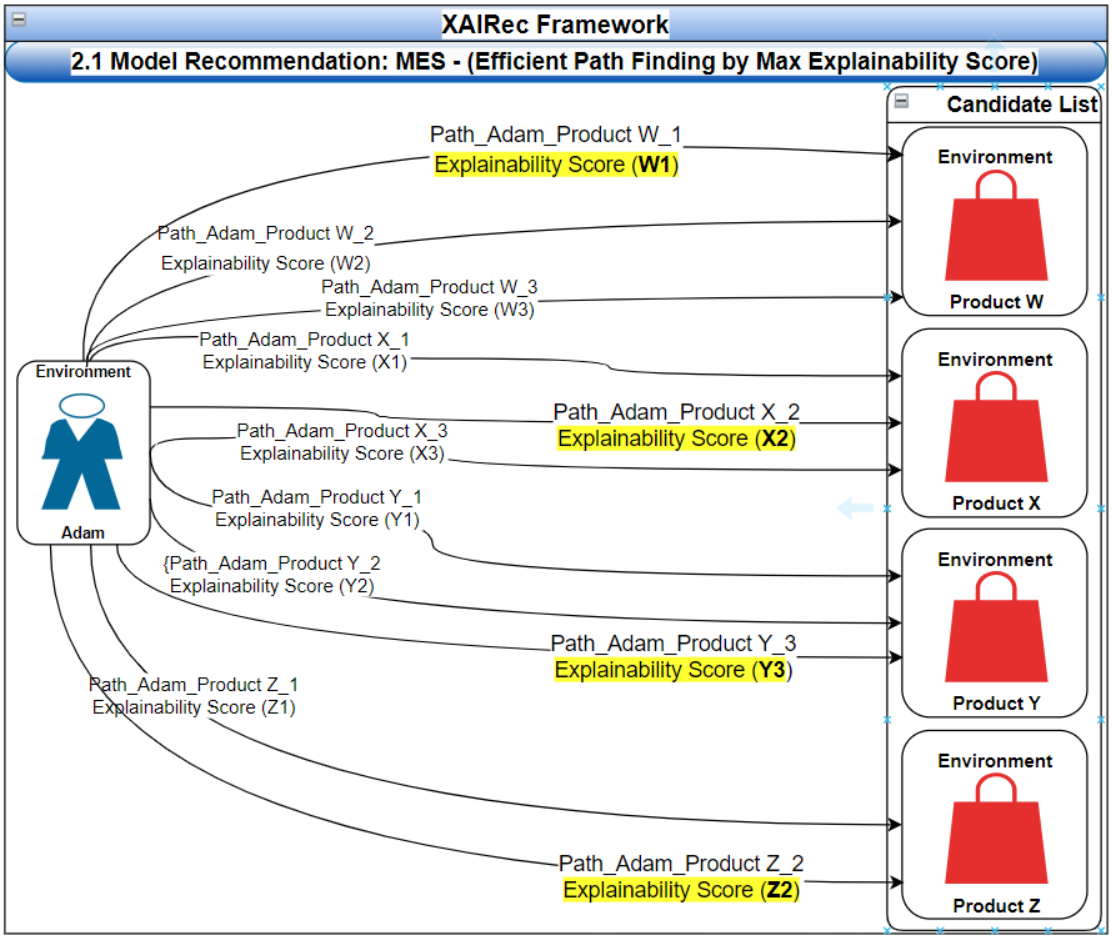

Figure 4.6 **Max Explainability Score – Efficient Pathfinding**

b. Illustrative Example

In **Figure 4.6**, the operational process of the MES algorithm is illustrated.

1. **Input Data and Context**: The process begins with a user named Adam and the context of the recommendation task. Adam's preferences and interactions within the system, learned during the RUPPEP phase, are considered to guide the explanation generation process.

2. **Potential Traversal Paths**: The algorithm assesses all potential traversal paths that link Adam to the candidate products under consideration for recommendation. These paths represent the various ways Adam might reach the products within the underlying data structure.

For instance, Adam may traverse to "Product W" through three different paths: "Path_Adam_Product_W_1," "Path_Adam_Product_W_2," and "Path_Adam_Product_W_3".

3. **Explainability Score Evaluation:** For each of these potential paths, the MES algorithm computes an explainability score, as suggested in Algorithm 2 and eq. 4.15. This score quantifies how effectively a given path can elucidate the rationale behind recommending a specific product to Adam.

   For the same instance shared in the previous step, the RL model may generate explainability scores for the previously identified traversal paths to Product W from user Adam as W1, W2, and W3.

4. **Identifying Optimal Path**: Among the evaluated paths, the MES algorithm identifies the traversal path with the highest explainability score. This path is highlighted in yellow to indicate its significance in providing a compelling and comprehensive explanation for the recommendation.

   In the same example, the highlighted highest explainability score is W1. This process is followed for all other candidate products as well.

5. **Meaningful and Diverse Explanation**: By selecting the traversal path with the highest explainability score, the algorithm ensures that the recommended product's rationale is conveyed to Adam in a meaningful and diverse manner.

6. **Enhanced Recommendation Experience**: The emphasis on optimal path contributes to the transparency and user-centric nature of the recommendation.

In essence, **Figure 4.6** captures the essence of the MES algorithm's functioning. It illustrates how the algorithm evaluates different paths, scores their explainability, and ultimately selects the most effective path for presenting recommendations to the user.

c. Algorithm - MES

Algorithm 2 generates a quantitative metric and provides the MES or $\boldsymbol{\mathcal{H}}^*$ of all the recommended products for users. It expects the inputs of users ($\boldsymbol{\mathcal{U}}$), items ($\boldsymbol{\mathcal{V}}$), and meta-graph paths ($\boldsymbol{\mathcal{P}_{MG}}$) in the KG ($\boldsymbol{\mathcal{G}_{\mathcal{XR}}}$) and generates the candidate set of explainabilities ($\boldsymbol{\mathcal{H}}$) for the recommended products. The algorithm outputs the optimal explanation of the recommended product for the user. It uses the embedding vectors of users, items, and their corresponding purchases. For the candidate product (v), it identifies all the probable paths ($\boldsymbol{\mathcal{P}_{MG}^{uv}}$) from the user ($u$) and evaluates the defined parameters $\mathbb{R}$, $\mathbb{P}$, $\mathbb{H}$, and $\mathbb{Rw}$. Every user is unique, and the mean values of the rewards ($\mathbb{Rw}_{User_{Mean}}$), probability ($\mathbb{P}_{User_{Mean}}$), and entropy ($\mathbb{H}_{User_{Mean}}$) for all the traversal paths of the recommended products for the users will also be different. Later, the algorithm evaluates differences in those parameters from the mean values ($\mathbb{Rw}_{diff_{User_{Mean}}}$, $\mathbb{P}_{diff_{User_{Mean}}}$, and $\mathbb{H}_{diff_{User_{Mean}}}$). The algorithm evaluates the $Rewards_{Gain}$ of the traversal path by taking a ratio of the sum of the rewards and the delta gain from the mean of the rewards values ($\mathbb{Rw} + \mathbb{Rw}_{diff_{User_{Mean}}}$) by the average of rewards values ($\mathbb{Rw} - \mathbb{Rw}_{diff_{User_{Mean}}}$) of the user for all the traversal paths of all the candidate recommended products and identifies the critical gain, as shown in Eq. 4.13.

$$\boldsymbol{Rewards_{Gain}} \leftarrow ((\mathbb{Rw} + \mathbb{Rw}_{diff_{User_{Mean}}}) / (\mathbb{Rw} - \mathbb{Rw}_{diff_{User_{Mean}}})) \quad \ldots(4.13)$$

Later, the algorithm evaluates the impact of $Entropy_{Gain}$ by summing up the entropy score with the delta gain from the mean entropy score of the user's traversal paths for all the recommended products. Entropy gain is a measure of new information or diversity a recommendation brings compared to the existing user choices. A higher entropy gain implies that the recommendation offers more novel or diverse options to the user.

$$\boldsymbol{Entropy_{Gain}} \leftarrow (\mathbb{H} + \mathbb{H}_{diff_{User_{Mean}}}) \quad \ldots(4.14)$$

The proposed method utilizes the benefits of both **valuable options** and **novelty or diversity** by multiplying the rewards gain with the entropy gain. This formulation suggests that an ideal recommendation not only provides valuable options ($Rewards_{Gain}$) but also introduces novelty or diversity ($Entropy_{Gain}$). The multiplied result divides by the number of rules ($\mathbb{R}$) to standardize the explainability score. The algorithm finally returns the explainable statement having maximum explainability.

$$\mathcal{H}_{Score} \leftarrow (\mathbf{Rewards}_{\mathbf{Gain}} * \mathbf{Entropy}_{\mathbf{Gain}}) \,/\, \mathbb{R} \qquad \ldots(4.15)$$

$$\mathcal{H}^{*} \leftarrow \mathbf{MAX}\,(\mathcal{H}_{\mathbf{Score}})$$

***Algorithm 2**: MES - Finding efficient paths through Max Explainability Score*

INPUT: A set of Users $\boldsymbol{\mathcal{U}}$, A set of Products or Items $\boldsymbol{\mathcal{V}}$, A set of Meta Graph Paths $\boldsymbol{\mathcal{P}_{MG}}$ in the KG, A candidate set of explainability $\boldsymbol{\mathcal{H}}$ of the recommended product for a user

OUTPUT: An optimal explanation $\boldsymbol{\mathcal{H}}^{*}$ of the recommended Product $\boldsymbol{\mathcal{V}}^{*}$

$u \leftarrow$ select $(\mathcal{U}, 1)$ // *select a user u from the user set* $\mathcal{U}$

$\mathcal{V}^{Purchase} \leftarrow$ *get_product (* $\mathcal{V}$*, u, 'Purchase')* // *get a list of purchased products of a user u.*

$\mathcal{V}' \leftarrow \{\mathcal{V} - \mathcal{V}^{Purchase}\}$ // *set of products, user u did not previously purchase.*

// create the embedding vectors

$\mathcal{U}_{Embed} \leftarrow Embed(\mathcal{U})$ // *the embedding vector of users*

$\mathcal{V}_{Embed}^{Purchase} \leftarrow Embed(\mathcal{V}^{Purchase})$ // *the embedding vector of purchases*

$\mathcal{U}\mathcal{V}_{Embed}^{Purchase} \leftarrow \mathcal{U}_{Embed} + \mathcal{V}_{Embed}^{Purchase}$ // *the embedding vector of user purchases*

$\mathcal{V}_{Embed} \leftarrow Embed(\mathcal{V})$ // *the embedding vector of products/items.*

// The logic of identification of all meta paths of user u to item v

for v **in** $\mathcal{V}'$: // v *is the current Product*

// Identify all the paths leading the user u to the product v

a. **for** $\mathcal{P}$ **in** $\mathcal{P}_{MG}$: // *probable paths from the user u to the product* v

// Traverse KG

b. **if** $\mathcal{P}$ **exists** between user u and item v**:**

i. $\mathcal{P}_{MG}^{uv} \leftarrow (u, v, \mathcal{P})$ // *Keep the identified paths*

// The basic explainability parameters for the meta paths of user u

for $\mathcal{P}$ **in** $\mathcal{P}_{MG}^{uv}$:

//1. Path Probability (*$\mathbb{P}$*) – path probability relates of user u to item v

// $\mathbb{P} \leftarrow Path(Probability) \leftarrow Path\big(Prob(nodes)\big)$

a. **for** $node$ **in** $\mathcal{P}$**:** // $node$ *is the current node in the path*

// the probability of the agent hopping from the current node to a

connected node defines as a RL model score based on the current environment $State$ *and the available* $Action$

i. $Prob(node) \leftarrow softmax\big(logits(State, Action)\big)$

b. $\mathbb{P} \leftarrow \prod_{i=1}^{n} Prob(node_i)$ *// n is the number of nodes in the path*

//2. Entropy ($\mathbb{H}$ ***) – Compute the entropy of the path relates user u to item v***

// $\mathbb{H} = Information\ Value(Path) = Entropy(Path)$

c. **for** *node* **in** $\mathcal{P}$**:** *// node is the current node in the path.*

d. $\mathbb{H} \leftarrow -\sum_{i=1}^{n} Prob(node_i) * log\ (Prob(node_i))$

//3. Reward ($\mathbb{R}w$ ***) – Compute the total RL framework reward for the path user u traverses to item v.***

e. **for** *node* **in** $\mathcal{P}$**:** *// node is the current node in the path.*

i. **if** len ($Path$) ≤ 2:

Reward ← 0

ii. **elseif** len ($Path$) ≥ 3 and not follows ***PATH PATTERN***:

Reward $\leftarrow -ve\ rewards$ *// Punish the agent with negative rewards.*

iii. **elseif** len ($Path$) ≥ 3 and follows ***PATH PATTERN***:

1. **if** type(*node*) == 'Product'

// dot product

a. ***Reward*** $\leftarrow \odot\ (\mathcal{UV}_{Embed}^{Purchase},\ \mathcal{V}_{Embed})$

// Maximize the rewards

b. ***Reward*** $*\leftarrow$ ***High +ve Reward***

2. **else** type(*node*) <> 'Product'

a. ***Reward*** $\leftarrow$ $matrix\_multiplication\ (\mathcal{U}_{Embed}, node_{Embed})$

b. ***Reward*** $*\leftarrow$ **+ve *Reward*** *// Encourage*

f. $\mathbb{Rw} \leftarrow \sum_{i=1}^{n} \boldsymbol{Reward}_{i}$

// Evaluate the mean scores of all the explainability parameters

10 **for** u **in** $\boldsymbol{\mathcal{U}}$: *// u is the current User*

//Evaluate the mean scores

// $\boldsymbol{n}$ is the number of products and m is number of paths in a product

i. $\mathbb{P}_{User_{Mean}} = \sum_{i=0}^{n}\sum_{j=0}^{m}\mathbb{P}_{ij} \Big/ (n+m)$ *// The mean probability of the traversed path for the user*

ii. $\mathbb{H}_{User_{Mean}} = \sum_{i=0}^{n}\sum_{j=0}^{m}\mathbb{H}_{ij} \Big/ (n+m)$ *// The mean entropy of the traversed path for the user*

iii. $\mathbb{Rw}_{User_{Mean}} = \sum_{i=0}^{n}\sum_{j=0}^{m}\mathbb{Rw}_{ij} \Big/ (n+m)$ *// The mean rewards of the traversed path for the user*

// Evaluate the difference of explainability parameters from the user means of those parameters

11 **for** $\boldsymbol{\mathcal{P}}$ **in** $\boldsymbol{\mathcal{P}_{MG}^{uv}}$:

//4. Rules ($\mathbb{R}$) – Identify the number of rules or features or hops.

a. $\mathbb{R} \leftarrow$ Count(nodes in $path\ \boldsymbol{\mathcal{P}}$) OR $Count(Hops\ in\ path\ \boldsymbol{\mathcal{P}})$

//5. Difference of explainability parameters from the user means.

b. $\mathbb{P}_{diff_{User_{Mean}}} \leftarrow \left(\mathbb{P} - \mathbb{P}_{User_{Mean}}\right)$ *// The diff of probability from the mean probability*

c. $\mathbb{H}_{diff_{User_{Mean}}} \leftarrow \left(\mathbb{H} - \mathbb{H}_{User_{Mean}}\right)$ *// The diff of entropy from the mean entropy*

d. $\mathbb{Rw}_{diff_{User_{Mean}}} \leftarrow \left(\mathbb{Rw} - \mathbb{Rw}_{User_{Mean}}\right)$ *// The diff of rewards from the mean rewards*

// Evaluation of explainability gains and explainability score for the path

12 **for** Exp in $\boldsymbol{\mathcal{H}}$**:**

a. ***Evaluate the below parameters.***

i. ***Evaluate*** $\mathbb{R}$, $\mathbb{P}$, $\mathbb{H}$, $\mathbb{Rw}$, $\mathbb{P}_{diff_{User_{Mean}}}$, $\mathbb{H}_{diff_{User_{Mean}}}$, $\mathbb{Rw}_{diff_{User_{Mean}}}$ **in** Exp

b. ***Evaluate the Rewards Gain***

i. $\mathbb{Rw}_{Gain} \leftarrow ((\mathbb{Rw} + \mathbb{Rw}_{diff_{User_{Mean}}}) / (\mathbb{Rw} - \mathbb{Rw}_{diff_{User_{Mean}}}))$

c. ***Evaluate the Entropy Gain***

i. $\mathbb{H}_{Gain} \leftarrow (\mathbb{H} + \mathbb{H}_{diff_{User_{Mean}}})$ *//Entropy gain*

// ***Evaluate the explainability score***

d. $\boldsymbol{\mathcal{H}_{Score}} \leftarrow (\mathbb{Rw}_{Gain} * \mathbb{H}_{Gain}) / \mathbb{R}$

// Evaluation of *max explainability score*

13 $\boldsymbol{\mathcal{H}}^* \leftarrow$ MAX ($\boldsymbol{\mathcal{H}_{Score}}$)

return $\boldsymbol{\mathcal{H}}^*$

d. Deciphering the Algorithm

Following is the elaboration of the steps involved in the algorithm.

1. Select a user *u* from the set of users $\{\boldsymbol{\mathcal{U}}\}$.

2. Get the set of purchased products $\{\boldsymbol{\mathcal{V}}^{Purchase}\}$ for this user *u* from the training dataset. These purchased products are the user's historical purchases.

3. Get the set of candidate products $\{\boldsymbol{\mathcal{V}'}\}$ for this user *u*, which can be potentially recommended to the user. These are the list of products after removing their earlier purchased products from the master list of products $\boldsymbol{\mathcal{V}}$.

4. The next step is to utilize the generated embedding vectors of users and product purchases to derive a new embedding vector of user product purchases $\boldsymbol{\mathcal{UV}}_{Embed}^{Purchase}$.

5. The next step is to find all the probable meta-paths of user *u* to item *v* belongs to $\{\boldsymbol{\mathcal{V}'}\}$ and keep in $\boldsymbol{\mathcal{P}}_{\boldsymbol{MG}}^{\boldsymbol{uv}}$.

6. From this point, the basic building blocks of the explainability evaluation process start. In this step, the algorithm evaluates the Path Probability ($\mathbb{P}$), Entropy ($\mathbb{H}$), Reward ($\mathbb{Rw}$), and number of hops or rules ($\mathbb{R}$) associated with the traversal paths. The algorithm evaluates these parameters for every path $\boldsymbol{\mathcal{P}}$ available in $\boldsymbol{\mathcal{P}}_{\boldsymbol{MG}}^{\boldsymbol{uv}}$.

7. Path Probability ($\mathbb{P}$) is the combined probability of the traversal path $\boldsymbol{\mathcal{P}}$, that connects user *u* to item *v*. $\mathbb{P}$ depends upon all the nodes that exist in the path $\boldsymbol{\mathcal{P}}$. The probability of the agent hopping from the current node to a connected node is defined as a RL model score based on the current environment $State$ and the available $Action$.

8. Entropy ($\mathbb{H}$) is the combined entropy of the traversal path $\boldsymbol{\mathcal{P}}$, that connects user *u* to item *v*. $\mathbb{H}$ depends upon all the nodes that exist in the path $\boldsymbol{\mathcal{P}}$ and can be evaluated with the earlier defined probabilities as described in the algorithm.

9. Reward ($\mathbb{Rw}$) is the combined rewards that the RL agent provides due to traversal over the path $\boldsymbol{\mathcal{P}}$, that connects user *u* to item *v*. $\mathbb{Rw}$ depends upon all the nodes that exist in the path $\boldsymbol{\mathcal{P}}$ and can be evaluated with the formula prescribed in the algorithm. The reward is the main function in the RL framework, and this algorithm rewards the agent's every action that leads them to find the optimal path.

10. Rules ($\mathbb{R}$) is the number of features or nodes that exist in the traversal path $\boldsymbol{\mathcal{P}}$.

11. After evaluating the basic building blocks, the next step is to find out the mean scores of these evaluated parameters for every user.

12. After evaluating the mean scores, the next step is to evaluate the difference of these explainability parameters from user means of those parameters.

13. After evaluating the difference of these explainability parameters from the user means of those parameters, the next step is to evaluate the Rewards Gain and Entropy Gain associated with the traversal path.

14. Rewards Gain is a ratio of the sum of the associated rewards with the delta gain from the mean of the rewards values by the average of rewards values of the user for all the traversal paths of all the candidate recommended products and identifies the critical gain. A higher reward gain indicates a more valuable recommendation to the user.

15. Entropy Gain is a ratio of the sum of the associated entropy score with the delta gain from the mean entropy score of the user's traversal paths for all the recommended products. Entropy gain is a measure of new information or diversity a recommendation brings compared to the existing user choices. A higher entropy gain implies that the recommendation offers more novel or diverse options to the user.

16. After evaluating the Rewards Gain and Entropy Gain, the algorithm multiplies those evaluated derived parameters and does a normalization by dividing with the number of Rules ($\mathbb{R}$) to get the explainability score ($\boldsymbol{\mathcal{H}_{Score}}$) associated with the traversal path ($\boldsymbol{\mathcal{P}}$) for the user *u* to item *v*.

17. The algorithm evaluates the explainability score for all the traversal paths ($\boldsymbol{\mathcal{P}}$) associated with the user *u* to item *v* and exist in $\boldsymbol{\mathcal{P}_{MG}^{uv}}$.

18. Finally, the algorithm evaluates the traversal path having the maximum explainability score $\mathcal{H}^*$, and returns that path and the corresponding explainability score.

The proposed algorithm utilizes the benefits of both **valuable options** and **novelty or diversity** by multiplying the rewards gain with the entropy gain. The algorithm proposes a new quantitative metric, MES (or $\mathcal{H}^*$), to measure recommendation explainability in the framework of KGs, RL, and embedded technologies.

4.4.3 Model Recommendation: PPS - Product Prioritization

This algorithm aids in product prioritization by arranging them in descending order based on their prioritisation score for the user. By doing so, it enhances the effectiveness of the RS by giving higher priority to products that are more appealing to users.

a. Products Prioritisation Score - Definition

RSs play a pivotal role in guiding users through the vast array of available products. However, due to the overwhelming number of options, it's essential to narrow down the suggestions to a select few that genuinely capture the user's interest. After identifying the most promising paths to potential products, the subsequent crucial phase involves ranking these products in a manner that optimally resonates with the user. To address this challenge, this study introduces an advanced technique known as the PPS (Eq. 4.17). This score serves as a metric, quantifying a product's priority for a specific user by carefully considering a multitude of pivotal factors. In practical terms, the RS leverages this PPS to make product suggestions that are not only aligned with the user's preferences and needs but are also more likely to establish a connection.

The essence of the PPS lies in its capacity to comprehensively assess and gauge the degree of priority a user holds towards a given product. This extends beyond conventional recommendation methods that might merely consider the popularity of items or rely solely on CF. The PPS introduces a more sophisticated

approach to product ranking and recommendation by encompassing an array of critical determinants.

Furthermore, the PPS technique seeks to balance recommending products that align with the user's known preferences and introducing an element of novelty or diversity. This prevents recommendation fatigue and enhances the user experience.

b. Illustrative Example

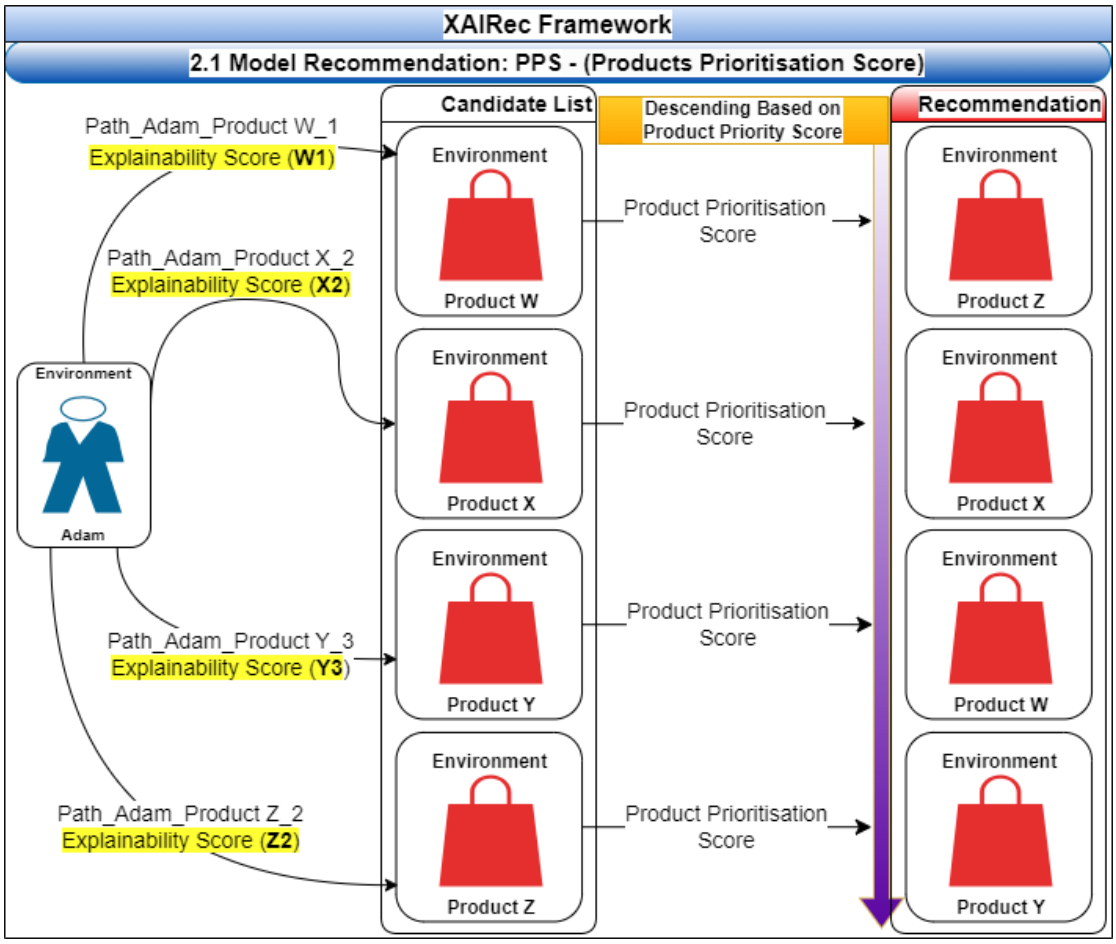

Figure 4.7 Products Prioritisation Score

Figure 4.7 illustrates the operational framework of the PPS algorithm (Eq. 4.17). The RS model, trained through RUPPEP component, identifies a set of products, namely, Product W, Product X, Product Y, and Product Z. Their respective explanatory paths,

highlighted in yellow, are determined through the implementation of the MES algorithm during the phase of product recommendation.

However, it's important to underscore that the list of candidate recommendable products could be voluminous, posing a formidable challenge during recommendation for the user Adam. This task, while pivotal, can be daunting given the vast array of available choices.

The PPS algorithm derives a prioritisation score for each candidate product, effectively gauging the degree of resonance based on the multiple derived parameters. This plays a valuable role in prioritizing the products for the user Adam.

By utilizing these prioritisation scores, the PPS algorithm undertakes the critical task of product prioritization. The algorithm effectively arranges the products in a descending sequence of their PPS values that reflect their level of alignment with the user's preferences. This sequence translates into a prioritized order of product recommendations presented to Adam.

Evidently, in the diagram, the products are sequenced based on their PPS scores. Specifically, Product Z emerges as the highest-priority recommendation, followed by Product X, Product W, and Product Y.

c. Algorithm - PPS

Algorithm 3 generates a quantitative metric and provides the PPS of all the recommended products for users. It expects the inputs of users ($\boldsymbol{\mathcal{U}}$) and items ($\boldsymbol{\mathcal{V}'}$) in the KG ($\boldsymbol{\mathcal{G}_{\mathcal{XR}}}$) and generates the prioritized recommended products. It uses the embedding vectors of users, items, and their corresponding purchases. For the candidate product ($\mathcal{v}$), it evaluates the $\mathbb{R}\mathbb{w}_{Gain}$, and $\mathbb{H}_{Gain}$ by using Algorithm 2. This algorithm considers the affinity score $\mathbb{S}$. Every user is unique, and the mean values of the affinity score $\mathbb{S}_{User_{Mean}}$ for the recommended products will also be different. Later, the algorithm evaluates the difference from the users' mean values $\mathbb{S}_{diff_{User_{Mean}}}$.

The algorithm assesses the $AffinityScore_{Gain}$ of the product by summing up the affinity score with the delta gain from the mean user affinity score for all the recommended products. Affinity score gain is a measure of user priority for the products. A higher affinity score gain implies that the recommendation entices users to take affirmative action.

$$\boldsymbol{AffinityScore_{Gain}} \leftarrow (\mathbb{S} + \mathbb{S}_{\boldsymbol{diff_{User_{Mean}}}}) \quad \ldots(4.16)$$

By summing the rewards gain with the affinity score gain and later multiplying it with the entropy gain, the proposed method aims to combine the benefits of all aspects. This formulation suggests that an ideal recommendation not only provides valuable options ($Rewards_{Gain}$), considering the utility of product recommendations ($AffinityScore_{Gain}$), but also introduces novelty or diversity ($Entropy_{Gain}$).

$$\boldsymbol{PPS_v^u} \leftarrow ((\mathbb{R}\mathbb{w}_{\mathbf{Gain}} + \mathbb{S}_{\mathbf{Gain}}) * \mathbb{H}_{\mathbf{Gain}}) \quad \ldots(4.17)$$

The algorithm ranks products based on their prioritisation score (PPS) in descending order. A high PPS indicates strong user interest. This score helps evaluate the effectiveness of the RS by prioritizing products that are more likely to be appealing to the user.

Algorithm 3: *PPS - Products prioritization based on Priority Score*

INPUT: A set of Users $\boldsymbol{\mathcal{U}}$, A set of candidate Items $\boldsymbol{\mathcal{V}'}$ in KG $\boldsymbol{\mathcal{G}_{\mathcal{XR}}}$

OUTPUT: Sorted prioritized Product recommendations

1 $u \leftarrow$ *select* ($\boldsymbol{\mathcal{U}}$, 1) *// A user u from the user set* $\boldsymbol{\mathcal{U}}$

2 $\boldsymbol{\mathcal{V}^{Purchase}} \leftarrow$ *get_product* ($\boldsymbol{\mathcal{V}}$, *u, 'Purchase') // Products: Train labels.*

3 $\boldsymbol{\mathcal{V}'} \leftarrow \{\boldsymbol{\mathcal{V}} - \boldsymbol{\mathcal{V}^{Purchase}}\}$ *// Products: User u did not previously purchase.*

// create the embedding vectors

4 $\boldsymbol{\mathcal{U}_{Embed}} \leftarrow Embed(\boldsymbol{\mathcal{U}})$ *// User Embedding*

5 $\boldsymbol{\mathcal{V}}^{Purchase}_{Embed} \leftarrow Embed(\boldsymbol{\mathcal{V}}^{Purchase})$ *// Purchase embedding.*

6 $\boldsymbol{\mathcal{UV}}^{Purchase}_{Embed} \leftarrow \boldsymbol{\mathcal{U}_{Embed}} + \boldsymbol{\mathcal{V}}^{Purchase}_{Embed}$ *// User purchases embedding.*

7 $\boldsymbol{\mathcal{V}_{Embed}} \leftarrow Embed(\boldsymbol{\mathcal{V}})$ *// Products embedding.*

// Evaluate all the parameters

8 *for* $\boldsymbol{v}$ ***in*** $\boldsymbol{\mathcal{V}'}$*:* *//* $\boldsymbol{v}$ *is the current Product.*

//1. ***Evaluate*** $\mathbb{R}$**,** $\mathbb{P}, \mathbb{H}$**,** $\mathbb{Rw}, \mathbb{Rw}_{Gain}, \mathbb{H}_{Gain}$ *through* **Algorithm 2.**

//2. ***Affinity Score (***$\mathbb{S}$***)*** *– Compute the affinity score for the product* $\boldsymbol{v}$*.*

9 $\mathbb{S} = Cosine(\boldsymbol{\mathcal{UV}}^{Purchase}_{Embed}, \boldsymbol{\mathcal{V}}^{\boldsymbol{v}}_{\boldsymbol{Embed}})$

// Evaluate the mean user scores for $\mathbb{S}$

10 *for* u ***in*** $\boldsymbol{\mathcal{U}}$*:* *//* $\boldsymbol{n}$ *= number of products*

11 $\mathbb{S}_{User_{Mean}} = {\sum_{i=0}^{n} \mathbb{S}_i}\Big/{(n)}$ *// User mean affinity score.*

// Evaluate differences of explainability parameters from the user means

12 *for* $\boldsymbol{v}$ ***in*** $\boldsymbol{\mathcal{V}'}$*:* *//* $\boldsymbol{v}$ *is the current Product.*

13 $\mathbb{S}_{diff_{User_{Mean}}} \leftarrow (\mathbb{S} - \mathbb{S}_{User_{Mean}})$ *// The diff of affinity score*

// Evaluation of affinity score gains

14 $\mathbb{S}_{Gain} \leftarrow (\mathbb{S} + \mathbb{S}_{diff_{User_{Mean}}})$

// Evaluate the Product Prioritisation Score

15 $PPS_v^u \leftarrow ((\mathbb{R}\mathrm{w}_{Gain} + \mathbb{S}_{Gain}) * \mathbb{H}_{Gain})$

// Return the Products Prioritisation Score of the product for the User.

16 **Return** PPS_v^u

d. Deciphering the Above Algorithm

Below is an elaboration of the steps involved in the algorithm:

1. Follow the steps outlined in MES's deciphering algorithm up to step 9, as enumerated in the previous section.

2. Evaluate the Affinity Score gain ($\mathbb{S}_{Gain}$) for all products in the candidate list for each user.

3. Derive the PPS by summing the Rewards Gain ($\mathbb{R}\mathrm{w}_{Gain}$) with the Affinity Score Gain ($\mathbb{S}_{Gain}$) and multiplying it with the Entropy Gain ($\mathbb{H}_{Gain}$) associated with the chosen traversal path.

4.4.4 Explainability – Quantitative Evaluation

RSs use various techniques to generate recommendations related to the user persona and the user's past purchase behaviour. In some situations, users might be interested in the explainability associated with these recommendations. The right explainability boosts the confidence and trust of the user in the system. However, the question lies in the methods to measure this explainability. There might be many ways to provide the explainability of those recommendations, but what constitutes good explainability? Out of all the possible explanations, which explanation is more suited to the end user? What methods could be used to measure the effectiveness of the various explainabilities? Qualitative studies assist in measuring the efficacies of these explainabilities, but conducting the same is not feasible in many circumstances. It

drives the demand for a robust quantitative approach to address the explainability measurement issue.

This study defines a new quantitative metric, MES (or $\mathcal{H}^*$), as described in Eq. 4.15, to measure recommendation explainability in the framework of KGs and RL technologies. After recommending the top products according to their PPS values, the XAIRec framework also provides an explainability score for these recommended products, generated based on the MES algorithm.

Algorithm 2 outlines the detailed process for quantitatively evaluating the generated explainability. This evaluation considers multiple factors, including the rewards gained due to the provided explainability and the inherent information value carried by the explainability. The algorithm involves a systematic scoring mechanism to assess the effectiveness of the generated explanations, which ensures a thorough assessment of the explainability's impact and usefulness within the given context.

4.5 SUMMARY

This chapter serves to unveil the intricate workings of the proposed XAIRec framework, a significant advancement within the realm of RSs. Central to this advancement is the integration of XAI principles, which empowers the framework to bridge the gap between machine learning complexity and user comprehensibility. Through this fusion, XAIRec not only delivers precise product recommendations but also provides users with a deeper understanding of the underlying recommendation process.

The chapter commences with a comprehensive overview of the framework, highlighting its pivotal role in achieving the objectives outlined in the initial chapter. By incorporating semantic technologies, XAIRec presents a novel approach to fulfilling the core research goals, emphasizing a user-centric perspective.

Furthermore, the chapter delves into the framework's design, presenting a detailed exposition of the RUPEP, MES, and PPS algorithms. This elucidation unveils

the underlying mechanics of these algorithms, bringing to light the advanced strategies that contribute to the framework's effectiveness.

The chapter is dedicated to unraveling the pivotal aspect of generating explainability associated with the recommendations. It outlines how the framework provides insights into the decision-making process, ensuring that users gain clarity on why certain recommendations are presented to them. Moreover, the chapter extends its coverage to the creation of a quantitative metric that measures the quality of explainability, adding a valuable dimension to the evaluation process.

Overall, this chapter provides a comprehensive exploration of the XAIRec framework. From its high-level objectives to the intricate details of algorithmic design and the critical importance of explainability, the chapter sheds light on the multi-faceted contributions that collectively make XAIRec a pioneering force in the realm of RSs.

CHAPTER V

MODEL EXPERIMENTATION

5.1 INTRODUCTION

This chapter delves into evaluating an XAI framework, XAIRec, using the Amazon e-commerce datasets. It provides detailed descriptions of the experiment carried forward to meet the research objectives. It discusses the experimental procedures and environment setup considered while developing the proposed XAI framework XAIRec, evaluating it against the four Amazon e-commerce datasets.

While few datasets posed computational challenges due to their size, the research study successfully executed the developed model on all four datasets. Notably, the model was evaluated over a substantial number of epochs, including cases where training was conducted for up to 100 epochs, demonstrating a rigorous effort to ensure the model's effectiveness across different scenarios.

The outline of this chapter is described as follows: Section 5.1 introduces the chapter's objectives and content flow. Section 5.2 elucidates the experimental setup, detailing hardware, software, and tools employed, ensuring transparency in the experimental context. In Section 5.3, the step-by-step methodology of the experiment is explained thoroughly, facilitating replication by other researchers. Finally, Section 5.4 offers a concise summary of the preceding content and serves as a transition to subsequent chapters, reinforcing fundamental aspects of the conducted experiment.

5.2 EXPERIMENTAL SETUP

This section delves deeper into the experimental setup conducted for the study.

5.2.1 Experimental Environment

The experimentation platform for this study revolves around utilizing Python as the primary programming language. Complementing Python, the experiment leverages Pytorch, a DL framework, to facilitate the implementation of various algorithms. The DL framework assists in the vision of the experiment's scope and incorporates KG, which serves as a structured representation of interconnected entities and relationships. Subsequently, it helps with the embedding generation for the graph entities and relationships into continuous vector spaces. Finally, it supports the RL Actor-Critic model development, serving as a powerful paradigm for making sequence-based decisions. By leveraging RL, the experiment aims to achieve semantic recommendations that align with the vision of XAI.

For a comprehensive overview of the experiment's specifics, detailed information can be found in **Table 5.1**, where the experimental settings are documented.

Table 5.1 Experimental Environment Settings

Environment	Type	Value	Version
H/W & OS	OS	Windows 10 Enterprise	10.0.19045
	Processor	12th Gen i7-12800H,	14 cores
	RAM	32 GB	
S/W & Programming	Programming Platform	PyCharm	2022.3.2 (CE)
	Programming Language	Python	3.9.16
	Deep Learning Framework	Pytorch	1.13.1
		tensorboardX	2.2

to be continued…

… continuation

DL Libraries	Torch	1.13.1
	Torchvision	0.14.1
ML libraries	Matplotlib	3.7.0
	Scipy	1.10.0
	Scikit-learn	1.2.1
	Statistics	1.0.3.5
Other Libraries	Argparse	1.4.0
	Certifi	2022.12.7
	Cycler	0.11.0
	Easydict	1.10
	Kiwisolver	1.4.4
	Numpy	1.23.5
	Pandas	1.5.3
	Pillow	9.4.0
	Protobuf	3.20.3
	Pyparsing	3.0.9
	Python-dateutil	2.8.2
	Pytz	2022.7
	Six	1.16.0
	Tabulate	0.9.0
	Threading	2.2.0
	Tqdm	4.64.1

5.2.2 Evaluation process

Usually, datasets are divided into training and test sets for evaluating a proposed model. The model uses the training set to develop and the test set to validate. Then,

the proposed model is evaluated based on the performance metrics to determine its accuracy.

This research study has big datasets comprising four different domain-specific datasets. This study divided the dataset into train and test datasets for all four domain-specific datasets. The users in both the train and test datasets are the same. The only variations are the products they purchased earlier and the feature words they mentioned for those products. This study covers this dataset division in detail in the later subsection.

The training dataset assisted in developing the KG, KGE, and later the Actor-Critic RL model based on the KGE dataset. The Actor-Critical RL model uses the test dataset to recommend the top-N future best products for the users in the test dataset.

Due to a larger user group size, this study ran the recommendation process in batches of 64 users. The model's efficacy is measured via the evaluation parameters suggested in the upper subsections.

Overall, the research approach incorporated the training-test split, KGE and RL model development, and batch recommendation to evaluate the performance of the RS on a large dataset. This methodology allows us to rigorously assess how well this model can provide relevant and effective recommendations and the explainability of the recommendations to users within these domains.

5.2.3 Model Development – Train and Test Datasets

This research divided the transaction dataset into training and test datasets, containing information on users' product purchases, mention of feature words, and product descriptions. **Table 5.2** presents statistical information about the mean and standard deviation of the train and test datasets used in the experiments.

Table 5.2 **Statistics of Train and Test Datasets for the Four Amazon e-commerce Datasets**

Relations		Description	CDs & Vinyl	Clothing	Cell Phones	Beauty
			Number of Records / Transactions			
Total Records		Number of transactions	1097591 (100%)	278677 (100%)	194439 (100%)	198502 (100%)
	Train	Training dataset	804090 (73.26%)	214696 (77.04%)	150048 (77.17%)	149844 (75.49%)
	Test	Test dataset	293501 (26.74%)	63981 (22.96 %)	44391 (22.83%)	48658 (24.51%)
			Statistics (Mean ± Standard Deviation) of Relations per Head Entity			
User $\xrightarrow{purchase}$ *Product*		Overall Statistics	14.58± 39.13	7.08± 3.59	6.97± 4.55	8.88± 8.16
	Train	Statistics of Train dataset	10.68± 27.39	5.45± 2.49	5.38± 3.16	6.7± 5.7
	Test	Statistics of Test dataset	3.9± 11.75	1.62± 1.13	1.59± 1.41	2.18± 2.48
User $\xrightarrow{mention}$ *Word*		Overall Statistics	2545.92± 10942.3	440.2± 452.38	652.08± 1335.76	806.89± 1344.08
	Train	Statistics of Train dataset	1846.88± 7667.51	338.1± 334.91	500.01± 979.78	605.01± 957.5
	Test	Statistics of Test dataset	699.04± 3284.36	102.1± 134.21	152.07± 382.55	201.88± 401.51
Product $\xrightarrow{described\_as}$ *Word*		Overall Statistics	2973.19± 5490.93	752.75± 909.42	1743.16± 3482.76	1491.16± 2553.93
	Train	Statistics of Train dataset	2157.47± 4024.57	578.23± 708.51	1336.77± 2698.4	1118.18± 1905.02
	Test	Statistics of Test dataset	890.14± 1614.42	201.19± 255.96	457.78± 892.98	419.34± 731.15

The split between the train and test datasets varies for the four datasets. Additionally, the table provides statistics on the mean and standard deviation of the frequencies of users purchasing products and mentions words related to the products. For example, in the "Beauty" dataset, a user purchases an average of 8.88 products

with a standard deviation of 8.16 number of products. The same statistics in the corresponding train and test datasets vary. In the training dataset, a user purchases an average of 6.7 products with a standard deviation of 5.7 number of products. In the test dataset, a user purchases an average of 2.18 products with a standard deviation of 2.48 number of products. Similarly, in the "Clothing" dataset, users purchase an average of 7.08 products with a standard deviation of 3.59. These statistics exhibit variations in the corresponding train and test datasets. In the training dataset, users purchase an average of 5.45 products with a standard deviation of 2.49, whereas, in the test dataset, users purchase an average of 1.62 products with a standard deviation of 1.13.

5.3 EXPERIMENTATION

This section outlines the comprehensive experimentation, as illustrated in **Figure 5.1,** conducted with the research datasets to achieve the objectives of this study.

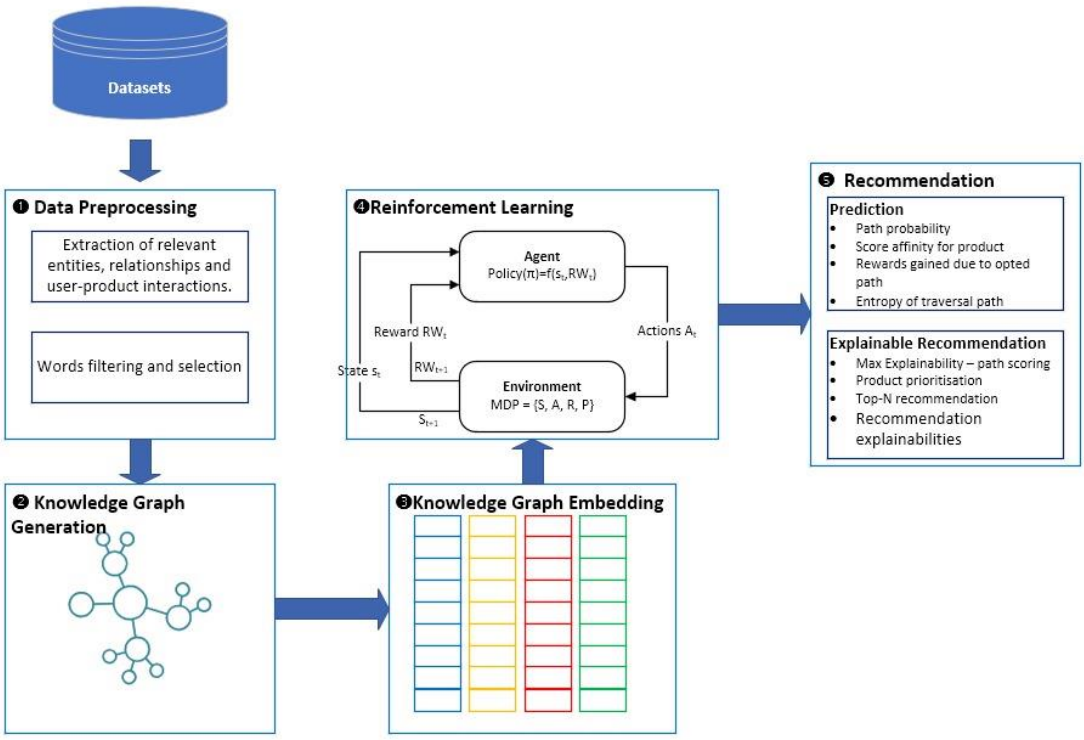

Figure 5.1 Experiment Architecture and Methodology

5.3.1 Beauty Dataset

This study starts the research work with the Beauty dataset and follows the steps mentioned below.

a. KG Formation

```
# KG - User nodes and their relationships
G = nx.Graph(kg(USER))
print(G)

# KG - Product nodes and their relationships
G = nx.Graph(kg(PRODUCT))
print(G)

# KG - Related Product nodes and their relationships
G = nx.Graph(kg(RPRODUCT))
print(G)

# KG - Category nodes and their relationships
G = nx.Graph(kg(CATEGORY))
print(G)

# KG - Brand nodes and their relationships
G = nx.Graph(kg(BRAND))
print(G)

# KG - Word nodes and their relationships
G = nx.Graph(kg(WORD))
print(G)
```

✓ 1.1s

```
Graph with 22365 nodes and 44726 edges
Graph with 12108 nodes and 84707 edges
Graph with 164724 nodes and 494163 edges
Graph with 249 nodes and 248 edges
Graph with 2078 nodes and 2077 edges
Graph with 22566 nodes and 45128 edges
```

Figure 5.2 Beauty KG – Nodes and Relationships

After preprocessing the dataset described in Section 3.5.2, the KG was generated according to the design outlined in Section 3.6.1. The KG consists of various nodes and relationships, as illustrated in **Figure 5.2**.

```
# KG - User nodes and their relationships of a particular user
G = nx.Graph(kg(USER, 12341))

plt.figure(figsize=(5,5))
pos = nx.spring_layout(G, k=0.6)
nx.draw(G, pos=pos, with_labels=True, node_color='red', edge_cmap=plt.cm.Blues, node_size=1000)
plt.show()
```

✓ 0.0s

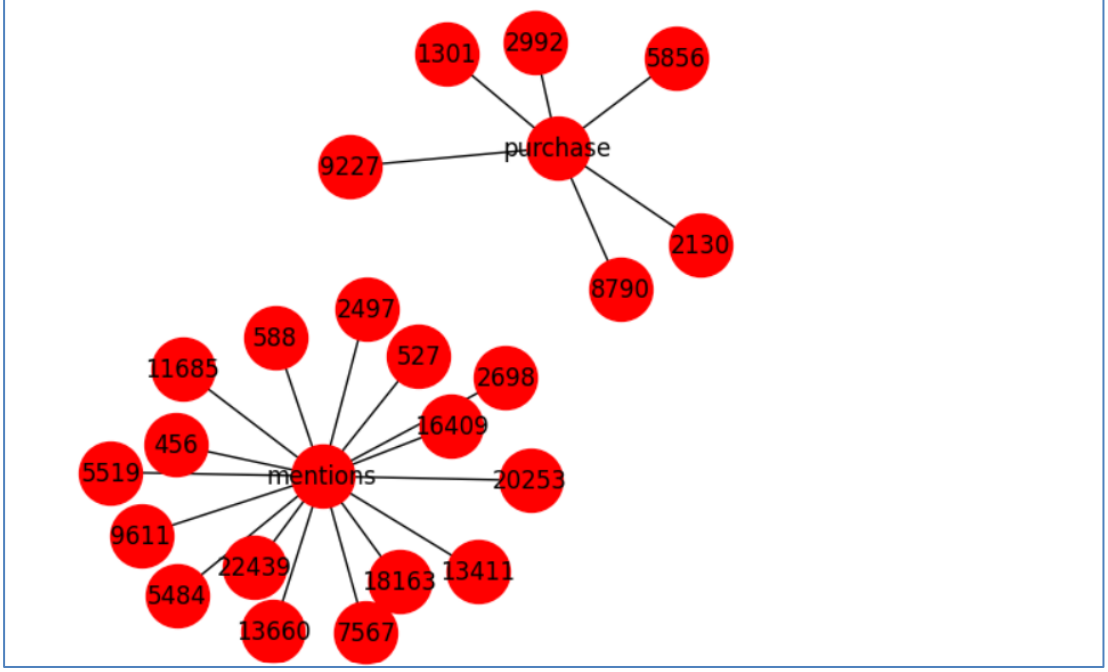

Figure 5.3 Beauty KG – A Relationship View for a Particular User

Figure 5.3 provides a detailed view of the relationships that a particular user (12341) has within the Beauty KG. This user has two types of relationships: "PURCHASE" and "MENTIONS." User 12341 has purchased several products and mentioned various product feature words while purchasing those products.

The subsequent nodes may have multiple relationships with various other nodes. **Figure 5.4** illustrates one such node, a product (9227). This product,

previously purchased by user 12341, has further relationships of the types: PURCHASE, DESCRIBE_AS, BELONGS_TO, PRODUCED_BY, ALSO_VIEWED, ALSO_BOUGHT, and BOUGHT_TOGETHER.

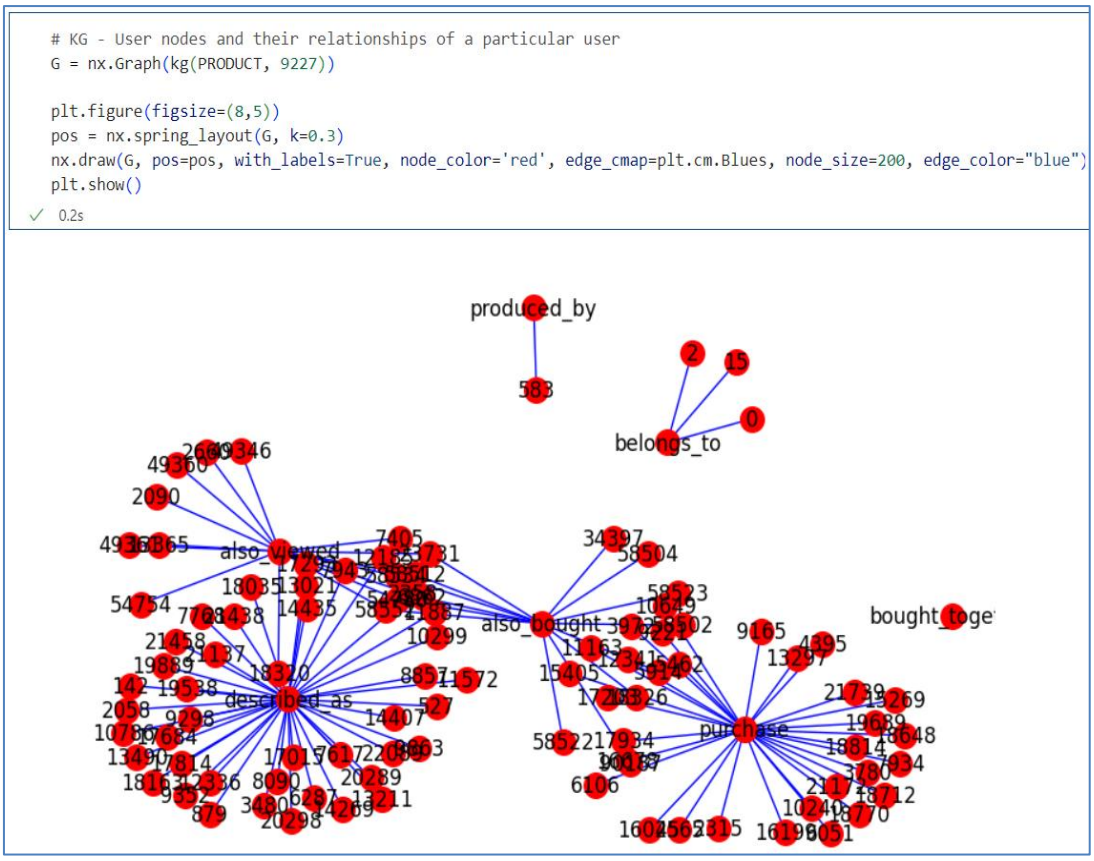

Figure 5.4 Beauty KG – A Relationship View for a Particular Product Purchased by the User

The product (9227) was purchased by a few other users and described using various feature words. It may be produced by a specific brand and belong to a few product categories. This product may also view with other related products. Sometimes, it may also buy with a few related products. Often, this product might be bought together, with a few other related products, but in this case, it is not bought together with any other products.

b. KGE Formation

After the Beauty KG's formation, the KG's embeddings were generated as outlined in section 3.6.2.

The following hyperparameters were used to develop the embeddings of the Beauty KG: the embedding model iterated for 30 epochs with a batch size of 64, and steps per checkpoint set at 10,000 to improve the generated embeddings. To prevent any loss of embedding generation efforts in case of failure, the model was designed to resume from the last checkpoint. Running for 30 epochs, the experiment iteratively improved the representation of entities and relationships, resulting in more efficient processing and increased accuracy in subsequent stages of the recommendation model. The embedding models continuously learned and updated the weights of the parameters, aiming to minimize the overall smooth loss associated with embedding.

The primary objective of the embedding model is to generate vector representations of the KG data. The aim is to ensure that these representations effectively capture the state of the KG, thereby minimizing any potential information loss during the embedding process. The KGE model plays a crucial role in this study, as its effectiveness directly impacts the quality of the recommendation model's results.

Table 5.3 outlines the chosen parameters for developing the KGE. Details regarding the execution of the KGE model can be found in the Appendix A.

Table 5.3 KGE Model – Parameters Selected Values

Parameters/Arguments	Search Space	Selected Value
dataset	One of {beauty, cd, cell, cloth}	BEAUTY
name	model name	train_transe_model
seed	Any numeric number	123
gpu	If GPU, then 1, else 0 for CPU	1
epochs	Positive numeric number	30
batch_size	Positive numeric number	64

to be continued…

… continuation

lr	float	0.5
weight_decay	Float for weight decay (L2 penalty) [0.0001, 0.001]	0.0001
l2_lambda	float	0
max_grad_norm	Positive number to address gradient exploding	5.0
embed_size	Positive numeric number	100
steps_per_checkpoint	Positive numeric number	10000
checkpoint_folder	Checkpoint folder name	checkpoint
is_resume_from_checkpoint	1 for resume from the last checkpoint else 0	1
num_neg_samples	Positive numeric number	5

c. RL Model Formation

After the formation of the KGE, the next step is to develop the RL model as outlined in section 3.6.3.

The following hyperparameters were used to train the RL model for the Beauty KG: the model was trained for 100 epochs with a batch size of 32, and checkpoints were saved every 5,000 steps to improve the generated RL model. To prevent any loss of progress in case of failure, the model was designed to resume from the last checkpoint. The learning rate was used as 0.0001 along with the maximum allowable actions as 250. The model was trained for a maximum of 3 hops. Two hidden layers used in this model with a size of 512 and 256.

Table 5.4 RL Model – Parameters Selected Values

Parameters/Arguments	Search Space	Selected Value
dataset	One of {beauty, cd, cell, cloth}	BEAUTY
name	model name	'train_RL_agent'
seed	Any numeric number	123
gpu	If GPU, then 1, else 0 for CPU	1

to be continued…

… continuation

epochs	Positive numeric number [1, 1000]	100
batch_size	Positive numeric number [16, 32, 64, 128, 256, 512, 1024]	32
lr	Float [0.0001, 0.001]	0.0001
max_acts	Positive numeric number [10, 1000]	250
max_path_len	Positive numeric number [1, 10]	3
gamma	Float value for reward discount factor [0, 1]	0.99
ent_weight	Float [0.0001, 0.01]	0.001
act_dropout	Float [0, 1]	0.5
hidden	Size of hidden layers [2^positive number]	[512, 256]
steps_per_checkpoint	Positive numeric number	5000
checkpoint_folder	Checkpoint folder name	checkpoint
is_resume_from_checkpoint	1 for resume from the last checkpoint else 0	1

The primary objective of the RL model is to train an RL agent based on the Beauty KG and its corresponding KGEs to capture users' path preferences and develop a policy that accurately predicts the next recommendation for users based on their prior purchases and other semantic relations. The goal is to minimize potential RL model loss during the training process by aligning the RL agent's actions with user preferences and awarding more rewards for following the set path patterns. The Actor-Critic RL model plays a crucial role in this study, as its effectiveness directly impacts the quality of the recommendation model's results.

Table 5.4 details the selected parameters for developing the RL model. The RL model generates checkpoint files that store the parameters of the effective Actor-Critic RL model, specifically ['l1.weight', 'l1.bias', 'l2.weight', 'l2.bias', 'actor.weight', 'actor.bias', 'critic.weight', 'critic.bias']. These weights are based on the multiple iterations that the RL models underwent during the training period. Details regarding the execution of the RL model can be found in Appendix A.

d. Model Evaluation

During the model evaluation phase, the model utilizes these checkpoint files to recommend products to the user. The performance of the model is then evaluated based on the accuracy and relevance of these recommendations.

The RL model is invoked for the test dataset with the following parameters. The model is tested using the last RL model checkpoint file from epoch 100, with a batch processing size of 512. The maximum allowable hops are set to 3. The test model utilizes the features of MES and PPS to further improve recommendation quality.

During each hop, nodes are trimmed based on the highest MES scores, with the allowable number of nodes set to [10, 10, 12]. This means that during the first hop, the maximum number of allowable nodes is set to 10. In the second hop, the allowable number of nodes remains 10, and in the third hop, it increases to 12. This trimming ensures that only the most relevant nodes are considered at each step, enhancing the precision of the recommendations.

Table 5.5 details the selected parameters during the recommendation phase. The test run resulted in final recommendations for the users, which were then evaluated against their true labels. Using the previously specified parameters, the model evaluation yielded the following test statistics. The model performed well during the test run, yielding the results as shown in **Table 5.6**.

Table 5.5 RL Model Recommendation – Parameters Selected Values

Parameters/Arguments	Search Space	Selected Value
dataset	One of {beauty, cd, cell, cloth}	BEAUTY
source_name	RL model name	'train_RL_agent'
output_folder	Output folder name	'test_RL_agent'
seed	Any numeric number	123
gpu	If GPU, then 1, else 0 for CPU	1

to be continued…

… continuation

epochs	Positive numeric number [1, 1000]	100
max_acts	Positive numeric number [10, 1000]	250
max_path_len	Positive numeric number [1, 10]	3
gamma	Float value for reward discount factor [0, 1]	0.99
hidden	Size of hidden layers [2^positive number]	[512, 256]
topk	number of permissible samples at every hop	[10, 10, 12]
run_path	Generate predicted path? True for Yes, and False for No.	True
run_eval	Run evaluation? True for Yes, and False for No.	True
Batch_size	Positive numeric number [16, 32, 64, 128, 256, 512, 1024]	512
checkpoint_folder	Checkpoint folder name	checkpoint
is_resume_from_checkpoint	1 for resume from the last checkpoint else 0	1
MES_score_option	Choose 0 for [Baseline], Choose 1 for [MES (Rewards Gain * Entropy Gain)], 2 for Only [Rewards Gain], 3 for Only [Entropy Gain], 4 for Only [Probs Gain], 5 for [Entropy Gain * Probs Gain], 6 for [Rewards Gain * Probs Gain], 7 for [Rewards Gain * Entropy Gain * Probs Gain], 8 for [Rewards Gain + Entropy Gain + Probs Gain]	1
PPS_score_option	Choose 0 for [Baseline], Choose 1 for [PPS ()], 2 for Only [Score], 3 for Only [Prob], 4 for Only [Entropy], 5 for [Reward]	1
is_only_run_specific_epoch	1 for specific epoch, and 0 for all epochs	1
dataset	One of {beauty, cd, cell, cloth}	BEAUTY

Table 5.6 Results from the Test Run

NDCG	Recall	Hit Ratio	Precision
6.62790	10.35920	17.21300	2.14212

These statistics demonstrate the model's performance in recommending products to users based on their prior purchases and other semantic relations. The precision, recall, NDCG, and hit ratio provide insights into how well the model predicted the true preferences of the users. Details regarding the execution of the RL model evaluation can be found in Appendix A.

5.3.2 Other Datasets

As described in section 5.3.1, similar steps were conducted for all other datasets, namely Clothing, Cell Phones, and CDs & Vinyl, in the study. Details of execution statistics can be found in Appendices B, C, and D, respectively.

Embedding vectors were generated in batches, utilizing a larger batch size of 512 user-product interactions for the CD KG due to its substantial size, while a batch size of 64 was used for other KGs.

5.4 SUMMARY

The experimentation chapter is crucial in the research study, as it is the practical testing ground for the proposed model or approach.

This chapter explains the outline of the experimental setup. This section encompasses the technical environment in which the experiments were conducted. It elucidates the specific computing infrastructure employed for the experiments, which, in this case, prioritized CPU processing over GPU due to certain constraints that imposed limitations on resource utilization.

The chapter then elucidates the choice of evaluation metrics adopted for the experimentation. These metrics assist in assessing the model's performance. Additionally, the chapter provides insight into the strategy employed for comparing the model's performance with baseline models, enabling a comprehensive assessment of its efficacy.

A pivotal aspect of the experimentation process is the delineation of the applied evaluation process. This process involves the division of the dataset into distinct training and testing splits, which serve as the foundation for conducting the experiments. This division is crucial in ensuring the separation of data for model development and subsequent evaluation.

In the final stages of the chapter, the research methodology utilized for the experiments is expounded upon. This methodology encompasses the development of KGE and RL models, which are integral to the proposed RS. Subsequently, the chapter details the evaluation mechanism of the RS's efficacies using evaluation metrics, providing a comprehensive view of its performance.

In summation, the experimentation chapter serves as the empirical backbone of the research study. It not only provides a practical framework for the proposed model but also offers transparency and rigor in the assessment of its performance.

CHAPTER VI

RESULTS AND DISCUSSIONS

6.1 INTRODUCTION

This chapter evaluates the XAIRec framework's performance on real-world datasets. The primary objective is to establish rigorous benchmarks to assess the efficacy of the XAIRec framework in comparison to other state-of-the-art models.

Through a systematic and empirical investigation, this study seeks to shed light on the strengths and weaknesses of the XAIRec framework, contributing to a deeper understanding of its practical utility. Furthermore, an essential aspect of this research is the impact analysis of parameter variations, including epochs, batch size, and the number of allowable samples for edge pruning within the method. The study aims to uncover insights into the robustness and versatility of the method.

This chapter is structured to explore the research on XR models comprehensively. It commences in Section 6.2 with a baseline comparison against state-of-the-art models, providing context for the advancements introduced by this model. Section 6.3 delves into ablation studies, evaluating the model's effectiveness across four Amazon e-commerce datasets. Section 6.4 deconstructs the core algorithms within the XAIRec framework, elucidating the generated explainabilities and highlighting the model's decision-making process through an illustrative example. Finally, Section 6.5 offers a summary of the chapter's key findings and contributions, allowing readers to grasp the core insights of the research.

6.2 XAIREC – BASELINE COMPARISON

6.2.1 Baselines Methods

This study compared the results of the proposed framework against several state-of-the-art approaches, as described in Chapter 2. The baselines include Collaborative Knowledge base Embedding (CKE) (Zhang W. et al., 2016), Joint Representation Learning (JRL) (Zhang Y. et al., 2017), Explainable Collaborative Filtering over Knowledge Graph (ECFKG) (Ai et al., 2018), Policy-Guided Path Reasoning (PGPR) (Xian et al., 2019), HR-RL-KG (Song et al., 2023), RKGR-RNS (Zhang et al., 2022), and Reinforcement learning framework for Multi-level recommendation Reasoning (ReMR) (Wang et al., 2022).

6.2.2 Model Comparisons

Through comparative analysis with state-of-the-art methods, this research establishes the effectiveness and superiority of the proposed approach in terms of recommendation performance and accuracy by applying four widely used top-N recommendation measures: NDCG, Recall, HR, and Precision. These metrics assess the ranking quality and effectiveness of the models in generating the top 10 recommendations for each user in the test set.

Table 6.1 presents the experimental results, demonstrating the performance of the XAIRec framework compared to all other baselines across multiple datasets. The metrics of NDCG, HR, Recall, and Precision consistently show significant improvements with XAIRec on various datasets. HR-RL-KG and RKGR-RNS were the two best baseline models based on RL technologies with embedded explanation generation methods. Although RKGR-RNS had good results compared to HR-RL-KG, it only provided performance results for three datasets (Beauty, Clothing, and Cell Phones). Another competitive baseline, ECFKG, was based on KGE technologies with a post-hoc explanation methodology. Notably, XAIRec achieved encouraging improvements of 5.16%, 2.65%, and 6.45% in NDCG on the Cell Phones, Beauty, and Clothing datasets, respectively, outperforming the best baseline (RKGR-RNS). These performance gains extend to other metrics, including Recall, HR, and Precision,

confirming the effectiveness of XAIRec in enhancing recommendation performance across diverse domains.

Table 6.1 Comparison of Effectiveness Metrics with Baseline Models.

Baseline Models	**CDs & Vinyl**				**Clothing**			
	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
CKE	4.620	6.483	14.541	1.779	1.502	2.509	4.275	0.388
JRL	5.378	7.545	16.774	2.085	1.735	2.989	4.634	0.442
ECFKG	5.563	7.949	17.556	2.192	3.091	5.466	7.972	0.763
PGPR	5.590	7.569	16.886	2.157	2.858	4.834	7.020	0.728
HR-RL-KG*[1]	5.70*	7.59*	16.94*	2.20*	3.14*	5.11*	7.39*	0.87*
ReMR	NA	NA	NA	NA	2.977	5.110	7.426	0.766
RKGR-RNS*[2]	NA	NA	NA	NA	3.10*	5.13*	7.47*	0.78*
XAIRec** (Ours)	**5.592**	**8.0345**	**17.855**	**2.3378**	**3.305**	**5.7059**	**8.2212**	**0.8655**
Improvement (%)	**-**	**5.8**	**5.43**	**6.36**	**6.45**	**11.31**	**10.04**	**11.54**

(*) represents the best baseline modes using RL (HR-RL-KG*1 – for all datasets, including CDs & RKGR-RNS*2 - for all datasets excluding CDs) and their published evaluation scores.

(**) represents the new proposed framework, XAIRec.

Improvement - % gain from the * baseline models

Baseline Models	**Cell Phones**				**Beauty**			
	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
CKE	3.995	7.005	10.809	1.070	3.717	5.938	11.043	1.371
JRL	4.364	7.510	10.940	1.096	4.396	6.949	12.776	1.546
ECFKG	5.370	9.498	13.455	1.325	6.399	10.411	17.498	1.986
PGPR	5.042	8.416	11.904	1.274	5.449	8.324	14.401	1.707
HR-RL-KG*[1]	5.43*	8.93*	12.50*	1.36*	5.83*	8.73*	14.78*	1.94*
ReMR	5.294	8.724	12.498	1.337	5.878	8.892	15.606	1.906
RKGR-RNS*[2]	5.81*	9.43*	13.24*	1.43*	6.42*	9.57*	16.25*	1.96*
XAIRec** (Ours)	**6.1086**	**10.460**	**14.678**	**1.599**	**6.59095**	**10.390**	**17.307**	**2.1577**
Improvement (%)	**5.16**	**10.92**	**10.88**	**11.89**	**2.65**	**8.57**	**6.46**	**10.20**

Path-based recommendation methods, including PGPR (Xian et al., 2019), ReMR (Wang et al., 2022), RKGR-RNS (Zhang et al., 2022), and HR-RLKG (Song et al., 2023) utilize RL as an MDP to address the issue of explaining entities that are more than one-hop away. These methods achieve this by searching for feasible paths between user-item pairs in the KG.

These methods target different aspects of explainability, user preference adaptation, and knowledge representation to suit varied application needs. However, challenges remain regarding personalization, optimal product recommendations, and identifying effective reasoning paths for these recommendations. They often fail to reward agents for exploring intermediary sub-paths that align with user preferences.

Additionally, discovering meaningful traversal paths on large graphs remains a challenge, impacting the overall accuracy of recommendations. Another significant challenge these methods face is prioritizing products for users based on their affinities towards those products.

The proposed framework, XAIRec, demonstrates enhanced performance compared to other baseline methods. This improvement is attributed to the integration of RUPPEP, MES, and PPS components within the XAIRec framework. These components facilitate personalised recommendations and address the crucial aspects of explainability and user affinity. As a result, XAIRec not only offers tailored suggestions but also ensures transparency and relevance in its recommendations to the user.

6.3 XAIREC – MODEL EVALUATION – ABLATION STUDY

6.3.1 Implementation Details

The default parameter settings for the experiments establish a consistent baseline for evaluating model performance. The latent representations of the models employ embeddings trained with a 1-hop scoring function, utilizing an embedding size of 100 and a learning rate of 0.5. Training the embeddings is limited to a maximum of 30 epochs. The $ActorCritic$ RL model prioritizes shorter paths for better user interpretation by considering a maximum path length of 3. The pruned action space has a maximum size of 250 actions, i.e., at most 250 actions for any state., with action dropout applied at a rate of 0.5 to encourage path diversity. The discount factor (γ) is set to 0.99. The policy/value network weights are defined as $W_1 \in \mathbb{R}^{400\times512}$, $W_2 \in \mathbb{R}^{512\times256}$, $W_P \in \mathbb{R}^{256\times250}$ and $W_V \in \mathbb{R}^{256\times1}$. During training, the models undergo 100 epochs using Adam optimization with a learning rate of 0.001 and a batch size of 64 for the CDs & Vinyl dataset, while for other datasets, the learning rate is 0.0001 with a batch size of 32. The weight of the entropy loss is 0.001. Additionally, both embeddings and RL models may resume from the point of the last checkpoint to ensure continuity and rerun in the experiments. In the model evaluation and product recommendation phase, the default parameters of the $ActorCritic$ model remain unchanged. The $ActorCritic$ model is evaluated for all users across the datasets used

in the experimentation, with a default sampling size parameter K for 3 hops: $K1 = 10$, $K2 = 10$, and $K3 = 12$. The evaluation phase incorporates novel methodologies of MES, and PPS to enhance the recommendations.

Table 6.2 Influence of Sampling Sizes on Recommendation Quality, with the Best Results Marked in Bold and the Default Experiment Setting Results Underlined, all Presented in (%).

Dataset	**Cell Phones**				**Beauty**			
Sampling Sizes	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
25,10,10	6.06791	10.266	14.344	1.5689	6.57605	10.332	17.213	2.1479
25, 10, 5	5.97740	10.022	14.151	1.5457	6.47486	10.152	16.882	2.0809
25, 10, 1	4.82123	7.4871	10.954	1.1772	5.16623	7.5205	13.277	1.5948
20,10,10	6.02041	10.239	14.323	1.5668	**6.59095**	**10.390**	**17.307**	**2.1577**
20,10, 5	5.95967	10.018	14.111	1.5414	6.44195	10.084	16.806	2.0701
20,10, 1	4.72446	7.3164	10.601	1.1352	5.06665	7.3279	12.915	1.5447
15,10,10	5.97637	10.141	14.233	1.5571	6.59256	10.283	17.047	2.1341
15,10,5	5.83862	9.8390	13.936	1.5184	6.42541	9.9639	16.587	2.0487
12,10,10	5.98970	10.184	14.305	1.5578	6.61043	10.304	17.101	2.1332
10,10,10	6.00657	10.180	14.244	1.5528	6.63440	10.318	17.136	2.1305
10,10,12	6.00481	10.223	14.301	1.5585	6.62790	10.359	17.213	2.1421
10,10,15	**6.02720**	**10.277**	**14.402**	**1.5682**	6.62122	10.332	17.146	2.1381

6.3.2 Impacts of Sampling Size Variations

In this experiment, this study investigated the influence of sampling size for path reasoning on the recommendation performance of the proposed framework. This study designed 12 combinations of sampling sizes, focusing on a path length or number of hops of 3. The results, reported in **Table 6.2**, showcased that the sampling sizes in the final hop of path reasoning had a more pronounced impact on finding effective paths and providing more prioritized product recommendations to end users. Interestingly, this study observes that the sampling sizes at the final hop play a more significant role in finding good product recommendations along with enhanced explainability. For example, in the cases of (25, 10, 10), (20, 10, 10), (15, 10, 10), (12, 10, 10), (10, 10, 10), (10, 10, 12), and (10, 10, 15), this model performs much better than in the rest of cases. This exhibit suggests that larger sample sizes at the final hop of path reasoning led to improved performance. Interestingly, after comparing the findings with other prominent baseline models that used similar RL approaches, as presented in **Table 6.3**, this study observed that the best sampling sizes for both PGPR and ReMR used a smaller sample size of 1 at the final hop compared to this model having a larger sample size and outperformed both the baseline models in evaluation metrics.

One explanation of this improvement is the adoption of two advanced algorithms: recommendation explainability (MES) and recommendation prioritization (PPS). After the first two hops are determined, the policy network converges toward selecting the optimal action for a desirable item mainly driven by the MES algorithm. Additionally, the model's stability with larger sample sizes at the last hop was noteworthy, providing valuable insights. This stability further underscored the effectiveness of the PPS algorithm, which played a role in prioritizing candidate products to enhance the recommendation process.

In contrast, baseline methods such as PGPR and ReMR utilize a sample size of 1 at the final hop, as they do not incorporate any product prioritization steps, which XAIRec does. By including the MES and PPS components, the XAIRec framework ensures a balance between exploration and exploitation when recommending products to users. The further experiment utilized the specified sampling setting of (10, 10, 12).

Table 6.3 **Comparison of the Influence of Sampling Sizes on Recommendation Quality, all Presented in (%).**

Best performing sampling size	Dataset	**Beauty**			
Competitive Models	*Sampling Sizes*	NDCG	Recall	HR	Precision
PGPR	10, 12, 1	5.926	9.166	15.667	1.920
ReMR	15, 10, 1	5.963	9.257	15.971	1.967
XAIRec	10, 10, 12	6.6279	10.359	17.213	2.1421

Overall, these findings highlighted the significance of sampling sizes and the efficacy of the novel algorithms in enhancing recommendation performance.

6.3.3 Impacts of MES and PPS

This study experimented to analyse the impact of MES and PPS on the effectiveness of the proposed framework. This study explored four different scenarios and measured their outcomes using various performance metrics. This study initially used the baseline setting and then applied different combinations of MES and PPS. The results, recorded in **Table 6.4**, demonstrated significant improvements in performance metrics across various datasets with the adoption of MES and PPS. The baseline method achieved an NDCG score of 1.58 on the Cell Phones dataset. While incorporating MES did not alter the score, integrating PPS resulted in a notable increase. By combining MES and PPS with the baseline model, the NDCG score improved even further. The overall NDCG gain of the final model compared to the baseline (i.e., without MES and PPS) was 280.4%. Notably, the XAIRec framework showed a remarkable 316.9% increase in NDCG on the Beauty dataset. The experiment's findings were consistent with other evaluation metrics, including Recall, HR, and Precision across multiple datasets. Overall, the experiment showcases the effectiveness of MES and PPS algorithms in providing explainable and improved recommendations for top products to users. MES facilitates product explanations, while PPS assists in product prioritization, ultimately enhancing the overall product recommendation process.

Table 6.4 Impact Analysis of MES and PPS on the Recommendation Performance

Dataset	**Cell Phones**				**Beauty**			
Parameters	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
Baseline (No MES & PPS)	1.57921	2.7960	3.9394	0.4205	1.59447	2.5543	4.9148	0.5925
Baseline + MES	1.57921	2.7960	3.9394	0.4205	1.59447	2.5543	4.9148	0.5925
Baseline + PPS	5.83965	9.9810	14.021	1.5230	6.45274	10.130	16.895	2.0983
Baseline + MES + PPS	**6.00481**	**10.223**	**14.301**	**1.5585**	**6.62790**	**10.359**	**17.213**	**2.1421**

The reason for not achieving any performance gain by adopting only the Baseline and MES approach is that MES solely provides the best path of explainability for a product to a particular user. It has no impact on performance unless combined with the PPS component. The PPS component prioritizes products based on users' affinities towards them, thereby enhancing the overall performance.

Table 6.4 also illustrates that the inclusion of both MES and PPS improves the efficacy of the framework, rather than the individual components alone. MES provides the optimal path with the maximum explainability score, thereby optimizing the explainability of the product recommendations, while PPS assists in prioritizing those recommendations based on user affinities.

6.4 EXPLAINABILITY EVALUATION

6.4.1 Explainability Evaluation

The main objective of this research is to evaluate the explainability of the recommended items. A user with previous product purchases might be interested in other relevant and exciting products. Upon receiving the recommendations through

the RS, the user might exclaim about the analogy of deriving such recommendations. The correct analogy may enthuse trust and confidence in the working methodology of the RS. Also, there could be many possible explanations for a recommended item, but which specific justification may be more suited to the end user? This research tried to answer these questions by experimenting with the Amazon dataset. Here the data values are encrypted to avoid resemblance of real-life products and users. This algorithm evaluates the explainability of all recommendations provided by the RS.

Since the model knows about the different traversal paths of the user for the recommended product in the KG, this algorithm tries to leverage those inherent properties to evaluate the importance of explainability. This algorithm evaluates the earlier defined parameters, $\mathbb{R}$, $\mathbb{P}$, $\mathbb{H}$, and $\mathbb{Rw}$. It assesses the explainability score $\mathcal{H}_{Score}$ based on those parameters for all the possible explanations for any recommendation. Finally, the algorithm returns the explainability with the maximum $\mathcal{H}_{Score}$, i.e., $\mathcal{H}^*$.

6.4.2 Explainability Illustration

In the context of a HIN KG, multiple paths can exist from the head entity to the target tail entity. It may present a challenge in providing explanations for product recommendations to users. In **Figure 6.1**, this study considers user 22012 [A3J5K0A02L0VG4] in the Beauty dataset, who received a product recommendation for the top 10 products. Among these, product 4266 [B005XP4YNQ] stands out. The user may desire an explanation for the reasoning behind this product recommendation.

Two potential explanations emerge from the KG:

- User 22012 [A3J5K0A02L0VG4] purchased product 8901 [B007PTGHKQ], which was also purchased by another user, 7790 [AFE0AGQT50MBJ]. User 7790 [AFE0AGQT50MBJ] had previously purchased product 4266 [B005XP4YNQ].

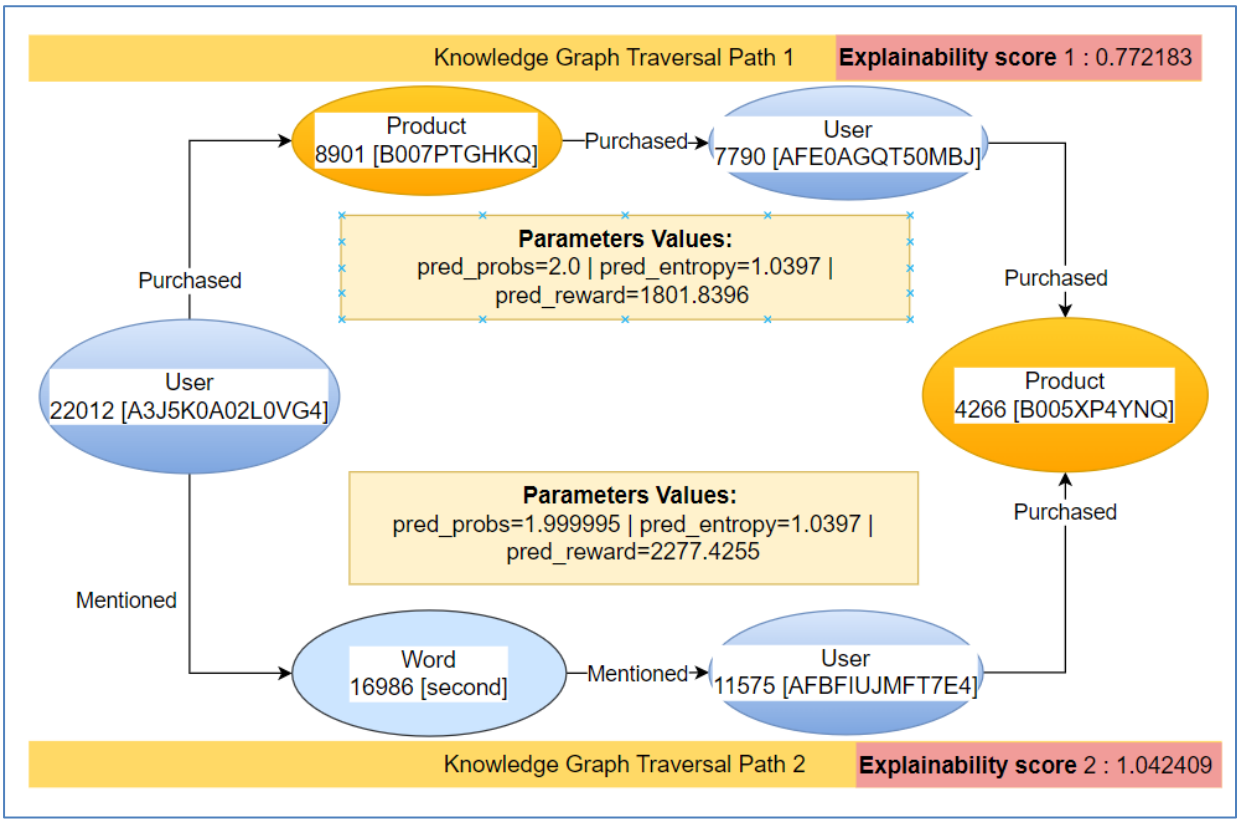

Figure 6.1 Explainability Illustration

- User 22012 [A3J5K0A02L0VG4] mentioned the word 16986 [second], and this word was also mentioned by user 11575 [AFBFIUJMFT7E4]. User 11575 [AFBFIUJMFT7E4] had purchased product 4266 [B005XP4YNQ].

Determining which explanation is most suitable is a challenge. However, the proposed framework resolves this ambiguity by calculating metrics and identifying the best-suited explainability for the end user. The proposed framework XAIRec generates an explainability score $\mathcal{H}_{Score}$ through **Algorithm 2** for both explanations based on predefined parameters $\mathbb{R}$, $\mathbb{P}$, $\mathbb{H}$, and $\mathbb{Rw}$. Based on these generated explainability scores, the algorithm returns the explainability with the highest score $\mathcal{H}^*$, i.e., the Maximal Explainability Score (MES).

In the given illustration, as traversal path 2 has the highest explainability score, the proposed framework would return that path as the explanation for the recommendation of product 4266 to user 22012. This ensures the provision of the

most suitable and explainable reasoning behind the product recommendation to the user.

To illustrate this, consider the following example scenario (**Figure 6.1**). The different possible explanations for this recommended product 4266 for user 22012 based on the KG traversals are:

- [('self_loop', 'user', 22012), ('purchase', 'product', 8901), ('purchase', 'user', 7790), ('purchase', 'product', 4266)]

User 22012 [A3J5K0A02L0VG4] has purchased product 8901 [B007PTGHKQ], which was also purchased by user 7790 [AFE0AGQT50MBJ], and user 7790 [AFE0AGQT50MBJ] had also purchased product 4266 [B005XP4YNQ].

- [('self_loop', 'user', 22012), ('mentions', 'word', 16986), ('mentions', 'user', 11575), ('purchase', 'product', 4266)]

User 22012 [A3J5K0A02L0VG4] has mentioned the word 16986 [second], and the word 16986 [second] was also mentioned by user 11575 [AFBFIUJMFT7E4], and user 11575 [AFBFIUJMFT7E4] had purchased a product 4266 [B005XP4YNQ].

Explainability Evaluation: *Quantitative Evaluation of a Product Recommendation Explainability for a User*

User Set ($\mathcal{U}$):

22012	A3J5K0A02L0VG4
22013	A3Q9XJFJLG07Y4

▷ *select a user u from the user set* $\mathcal{U}$ **, let the user is 22012**

1 $u \leftarrow$ select $(\mathcal{U}, 1)$

▷ *get a list of past purchased products of a user u*

2 $\mathcal{V}^{Purchase} \leftarrow$ *get_product* ($\mathcal{V}$, *u, 'Purchase')*

Open

Snippet of user 22012's past purchased products:

```
train_labels[uid] : 22012 [4996, 8901, 8442, 7176]
```

The top 10 recommended products by the model for the user:

```
pred_labels[uid] : 22012 [10369, 1054, 270, 7878, 9781, 6387, 6650, 4266, 3720, 1863]
```

The explainability for the 10 recommended items for user 22012:

```
22012 (10369, 'user 22012 has purchase product 8442 which was purchase by user 19233 who purchase product 10369')
explainability_score:  1.5695034160824113
22012 (1054, 'user 22012 has mentions word 16986 which was mentions by user 11575 who purchase product 1054')
explainability_score:  1.5325436735544966
22012 (270, 'user 22012 has purchase product 8901 which was purchase by user 13083 who purchase product 270')
explainability_score:  1.3287991679501054
22012 (7878, 'user 22012 has purchase product 4996 which was purchase by user 10173 who purchase product 7878')
explainability_score:  1.4289923067806463
22012 (9781, 'user 22012 has purchase product 4996 which was purchase by user 10173 who purchase product 9781')
explainability_score:  1.3105489617661314
22012 (6387, 'user 22012 has mentions word 6403 which was mentions by user 1930 who purchase product 6387')
explainability_score:  1.3126411470015875
22012 (6650, 'user 22012 has purchase product 8901 which was purchase by user 7790 who purchase product 6650')
explainability_score:  0.8926247823091165
22012 (4266, 'user 22012 has mentions word 16986 which was mentions by user 11575 who purchase product 4266')
explainability_score:  1.0424090993222799
22012 (3720, 'user 22012 has purchase product 8442 which was purchase by user 10069 who purchase product 3720')
explainability_score:  0.9407373809265469
```

Figure 6.2 An Example of Explainability

The first explanation, "User 22012 [A3J5K0A02L0VG4] has purchased product 8901 [B007PTGHKQ], which was also purchased by user 7790

[AFE0AGQT50MBJ], and user 7790 [AFE0AGQT50MBJ] had also purchased product 4266 [B005XP4YNQ].", yields the below metrics for recommended parameters.

Here $\mathbb{R}$ = 4, $\mathbb{P}$ **= 2.0,** $\mathbb{H}$ = **1.0397207736968994**, $\mathbb{R}w$ **= 1801.8396526575089,**

$\mathbb{P}_{diff_{User_{Mean}}}$ **=0.3728814731209964,**

$\mathbb{H}_{diff_{User_{Mean}}}$ **= -0.03327432769840044,**

$\mathbb{R}w_{diff_{User_{Mean}}}$ **=916.1858923717449**

$\mathbb{R}w_{Gain}$ = $((\mathbb{R}w + \mathbb{R}w_{diff_{User_{Mean}}}) / (\mathbb{R}w - \mathbb{R}w_{diff_{User_{Mean}}}))$ = **3.068948**

$\mathbb{H}_{Gain} \leftarrow (\mathbb{H} + \mathbb{H}_{diff_{User_{Mean}}})$ = **1.00644**

$\boldsymbol{\mathcal{H}_{Score}} \leftarrow (\mathbb{R}w_{Gain} * \mathbb{H}_{Gain}) / \mathbb{R}$

$\boldsymbol{\mathcal{H}_{Score}} \leftarrow$ **0.772183**

The second explanation, "User 22012 [A3J5K0A02L0VG4] has mentioned a word 16986 [second], and the word 16986 [second] was also mentioned by the user 11575 [AFBFIUJMFT7E4], and user 11575 [AFBFIUJMFT7E4] had purchased a product 4266 [B005XP4YNQ].", yields the below metrics for recommended parameters.

Here $\mathbb{R}$ = 4, $\mathbb{P}$ **= 1.999995,** $\mathbb{H}$ = **1.039721**, $\mathbb{R}w$ **= 2277.426,**

$\mathbb{P}_{diff_{User_{Mean}}}$ = **0.372877,**

$\mathbb{H}_{diff_{User_{Mean}}}$ = **-0.03327,**

$\mathbb{R}w_{diff_{User_{Mean}}}$ = **1391.77**

$\mathbb{R}w_{Gain}$ = $((\mathbb{R}w + \mathbb{R}w_{diff_{User_{Mean}}}) / (\mathbb{R}w - \mathbb{R}w_{diff_{User_{Mean}}}))$ **= 4.142925**

$\mathbb{H}_{Gain} \leftarrow (\mathbb{H} + \mathbb{H}_{diff_{User_{Mean}}})$ = **1.006447**

$\boldsymbol{\mathcal{H}_{Score}} \leftarrow (\mathbb{R}w_{Gain} * \mathbb{H}_{Gain}) / \mathbb{R}$

$\boldsymbol{\mathcal{H}_{Score}} \leftarrow$ **1.042409**

$\mathcal{H}^* \leftarrow$ MAX ($\mathcal{H}_{Score}$) = **1.042409**

So, the final recommended explainability for the recommended product would be the second explanation. A similar analogy will apply to all the recommended products for all users. **Figure 6.2** also provides another finding that the ***high explainability score products (shaded in blue colour) are at the top of the recommended list than the low explainability score products***. It validates the hypothesis of the impact of explainability score on the ranking and recommendation of products.

6.4.3 Impact of Explainability in Model Evaluations

The KG consists of many inter-connected dense paths from users to different entities. Referring to the previous example, **Table 6.5** summarises 147 different paths that the user 22012 [A3J5K0A02L0VG4] may take to reach product 4266 [B005XP4YNQ]. It is just one example of the number of possible paths that may be there in the KG for a product from the user. Similarly, the user might have many relations with many different products too. On the same note, since the KG has many users, it will have humongous possible paths.

So, the real question is how the recommendation model recommends the top 10 products out of many available products with many possible meta-paths. This experiment tried to address this concern by applying RL over the KG and applying the MES to prioritise the most explainable path that the user may opt for the product.

Table 6.5 Statistics of the Number of Paths for a User-Product Path Example

User	Product	Pattern	Number of Paths
22012 [A3J5K0A02L0VG4]	4266 [B005XP4YNQ]	((USER), (MENTION, WORD), (DESCRIBED_AS, PRODUCT))	1
		((USER), (MENTION, WORD), (MENTION, USER), (PURCHASE, PRODUCT))	6
		((USER), (PURCHASE, PRODUCT), (PURCHASE, USER), (PURCHASE, PRODUCT))	1

to be continued…

… continuation

((USER), (PURCHASE, PRODUCT), (DESCRIBED_AS, WORD), (DESCRIBED_AS, PRODUCT))	14
((USER), (PURCHASE, PRODUCT), (PRODUCED_BY, BRAND), (PRODUCED_BY, PRODUCT))	0
((USER), (PURCHASE, PRODUCT), (BELONG_TO, CATEGORY), (BELONG_TO, PRODUCT))	6
((USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (ALSO_BOUGHT, PRODUCT))	83
((USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (ALSO_VIEWED, PRODUCT))	19
((USER), (PURCHASE, PRODUCT), (ALSO_BOUGHT, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT))	0
((USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (ALSO_BOUGHT, PRODUCT))	9
((USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (ALSO_VIEWED, PRODUCT))	8
((USER), (PURCHASE, PRODUCT), (ALSO_VIEWED, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT))	0
((USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (ALSO_BOUGHT, PRODUCT))	0
((USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (ALSO_VIEWED, PRODUCT))	0
((USER), (PURCHASE, PRODUCT), (BOUGHT_TOGETHER, RPRODUCT), (BOUGHT_TOGETHER, PRODUCT))	0
Total Paths	**147**

Recommending the top 10 products has two primary steps. Firstly, the identification of the optimal path for the candidate product. Later, sorting those

products from the pool of selected candidate products and subsequently recommending them to the user. The optimal path for the candidate product not only assists in explaining the reasoning behind the recommendations with satisfactory explanations but also increases the chances of that product being under the top 10 recommendations category for the user. Thus, explainability drives better recommendations.

To understand the reasoning behind the explainability score formulation as a method of multiplying the Rewards gain with the Entropy gain ($\mathbb{R}\mathbb{w}_{Gain} * \mathbb{H}_{Gain}$), as prescribed in this thesis, **Table 6.6** describes the experimental results and compares the model recommendation performance metrics with different path priority algorithms. The experiment was performed on two datasets, namely "Beauty" and "Clothing", and evaluated using four metrics, namely NDCG, Recall, HR, and Precision. The experiment compared the impact of different path priority algorithms and explainability score formulations on the model evaluation metrics. For example, in the “Beauty” dataset, the $\mathbb{R}\mathbb{w}_{Gain}$ path priority approach generated an NDCG score of 6.3706, a Recall score of 10.2882, an HR score of 17.215, and a Precision score of 2.13586, whereas the $\mathbb{H}_{Gain}$ path priority approach produced an NDCG score of 6.68497, a Recall score of 10.302, an HR score of 17.2622, and a Precision score of 2.14346. The $\mathbb{R}\mathbb{w}_{Gain}$ indicates the high values associated with the traversal paths of the recommended products. It also refers to the improvement in the user's satisfaction or utility achieved by following a recommended path. The higher the rewards gain, the more beneficial the recommendation is considered. The $\mathbb{H}_{Gain}$ indicates the diversity or new information due to the traversal paths of recommended products provided to the user.

Table 6.6 Recommendation Effectiveness Metrics in Percentage (%) based on Top-10 Predictions in the Test Set.

Baseline Models	Clothing				Beauty			
	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
1. Baseline Model - With No Explainability Metrics applied	2.66976	4.60029	6.83999	0.70837	6.49722	10.1203	17.00282	2.10545
2. Baseline Model - With ($\mathbb{P}_{Gain}$) for path priority	2.66090	4.59647	6.82476	0.70431	6.50748	10.1031	16.9626	2.10098

to be continued…

… continuation								
3. Baseline Model - With ($\mathbb{R}\mathrm{w}_{Gain}$) for path priority	2.73371	4.68603	6.97456	0.72031	6.63706	10.2882	17.2175	2.13586
4. Baseline Model - With Entropy Gain ($\mathbb{H}_{Gain}$) for path priority	2.72941	4.65351	6.91870	0.71574	**6.68497**	10.3002	**17.2622**	2.14346
5. Baseline Model - With ($\mathbb{H}_{Gain} * \mathbb{P}_{Gain}$) Gain	2.66141	4.59647	6.82476	0.70431	6.50748	10.1031	16.96257	2.10098
6. Baseline Model - With ($\mathbb{R}\mathrm{w}_{Gain} * \mathbb{P}_{Gain}$) Gain	2.67710	4.62785	6.86284	0.71117	6.53163	10.1401	17.00282	2.10590
7. Baseline Model - With ($\mathbb{R}\mathrm{w}_{Gain} + \mathbb{P}_{Gain} + \mathbb{H}_{Gain}$) Gain	2.69229	4.62836	6.88062	0.71294	6.54818	10.1565	17.02965	2.10903
8. Baseline Model - With ($\mathbb{R}\mathrm{w}_{Gain} * \mathbb{P}_{Gain} * \mathbb{H}_{Gain}$) Gain	2.67710	4.62785	6.86284	0.71117	6.53077	10.1402	17.0028	2.10590
9. Baseline Model - With ($\mathbb{R}\mathrm{w}_{Gain} * \mathbb{H}_{Gain}/\mathbb{R}$) = (MES or $\boldsymbol{\mathcal{H}}^{*}$)	**2.74793**	**4.68726**	**6.98218**	**0.72183**	6.67281	**10.3177**	17.2488	**2.14346**

Higher entropy gain implies that the recommendation offers new and diverse options to the user. The multiplication of the $\mathbb{R}\mathrm{w}_{Gain}$ with the $\mathbb{H}_{Gain}$ implies that the generated recommendations offer both valuable options in terms of utility due to $\mathbb{R}\mathrm{w}_{Gain}$ and novelty or diversity due to $\mathbb{H}_{Gain}$. The MES, formulated as the multiplication of ($\mathbb{R}\mathrm{w}_{Gain} * \mathbb{H}_{Gain}/\mathbb{R}$) generated an NDCG score of 6.67281, a Recall score of 10.3177, an HR score of 17.2488, and a Precision score of 2.14346. The highest values derived in those metrics by different variations of the explainability score formulations are highlighted in bold. Recall and Precision are more important metrics than Hit Ratio and NDCG for evaluating the performance of a recommendation engine. In the experiment results, the ($\mathbb{R}\mathrm{w}_{Gain} * \mathbb{H}_{Gain}/\mathbb{R}$) formulation yielded the best values of Recall (10.3177) and Precision (2.14346) metrics for the Beauty dataset.

The baseline recommendation model indicates the approach without applying any path priority algorithm for choosing the optimal path associated with a product for a user. The baseline model generates an NDCG score of 6.49722, a Recall score of 10.1203, an HR score of 17.00282, and a Precision score of 2.10545. The MES formulation approach yielded an increase of 2.70254% NDCG, 1.950535% Recall, 1.446701% Hit Ratio, and 1.805315% in Precision scores than the baseline method in the Beauty dataset. The experiment results, and optimal formulations observed in the "Beauty" dataset were also observed and replicated in the "Clothing" dataset, indicating consistent results across different datasets. The results show that both $\mathbb{R}\mathbb{w}_{Gain}$ and $\mathbb{H}_{Gain}$ parameters are essential for explainability, and the highest values for the evaluation metrics were obtained with formulations that included both parameters.

As described earlier, the recommendation steps used in this experiment have two primary steps. The first step is to decide the optimal path for the user to the product. This optimal path will also explain the reasoning for the recommendation of the product to the user. Strong reasoning yields high explainability, and in turn, high explainability produces a high explainability score. Thus, the explainability score leads to the optimal path selection approach for the user to a product. The real question is how to form the explainability scoring method. As mentioned earlier, the explainability will be high if the information value provided through that explainability is higher, and in turn, the entropy of the path is higher. A high explainability should also yield a high reward. Thus, explainability is a function of gain due to rewards, and entropy. The overall high explainability score may also positively impact the effectiveness metrics of the recommendation engine.

The same is evident from the comparative experiment results produced here. The experiment results confirm the increase in model evaluation metrics of all the parameters with the adoption of the MES or $\boldsymbol{\mathcal{H}}^*$ metric in the model test pipeline.

6.5 SUMMARY

In this chapter, the focus has been on delving into the multifaceted aspects of the research about XR models. The chapter's structured approach allowed the study to explore and present the findings systematically.

The study was initiated with a baseline comparison, a critical starting point in evaluating the efficacy of the XR model. By pitting the proposed model against the state-of-the-art in the field, this study provided a solid benchmark, revealing the extent of advancements and improvements that this model brings to the realm of RSs. This comparison helps establish the significance of the research within the broader context of RSs. This model performed well against the existing state-of-the-art models, notable from recent research conducted by (Xian et al., 2019), and (Wang et al., 2022).

Moving onward to Section 6.3, this study embarked on a detailed examination of the model's performance through ablation studies. These studies were conducted across four distinct Amazon e-commerce datasets, enabling to discern the individual contributions of various components within the model. This in-depth analysis sheds light on which elements are most pivotal in driving the model's effectiveness. By conducting these studies, it offers valuable insights for researchers and practitioners seeking to optimize their RSs.

In Section 6.4, this study navigated deeper into the core algorithms at the heart of the XAIRec framework. This section was dedicated to elucidating the explainabilities generated by the framework. This study provided a granular understanding of how this model arrives at recommendations and the rationale behind those recommendations. To make this process more tangible, this study employed a concrete example, offering readers a direct illustration of the entire explainability generation process. This aids in comprehending the model's inner workings and contributes to the broader discourse on transparent and interpretable RSs.

In essence, this chapter has been a comprehensive journey through the intricacies of the research, providing a foundation for understanding the significance of the XR

model, its components, and its potential impact on the field of RSs. Through rigorous evaluation, in-depth analysis, and illustrative examples, this study aims to contribute to the ongoing discourse on the transparency and effectiveness of RSs.

CHAPTER VII

CONCLUSION AND FUTURE WORK

7.1 INTRODUCTION

This chapter consolidates the research discoveries, discusses their broader significance, and outlines avenues for future exploration. Beyond summarizing the outcomes achieved, the chapter aims to underscore the enduring relevance and potential for further advancements in the realms of this study. This chapter provides a bridge that connects the present accomplishments to the intriguing possibilities that lie ahead.

This chapter follows a structured approach: Section 7.2 summarizes the research findings and outcomes, shedding light on their significance and potential implications, while Section 7.3 explores the research's valuable contribution to the realm of XAI, particularly in the context of enhancing transparency in RSs. Section 7.4 outlines prospective directions for future research in this domain, identifying unexplored avenues and opportunities for further investigation. Finally, Section 7.5 provides a comprehensive conclusion, summarizing the chapter's key takeaways and emphasising the research's broader impact on the fields of XAI and RSs.

7.2 RESEARCH CONCLUSION

In conclusion, this study has contributed to overcoming the limitations of transparency and interpretability in RSs through a novel approach named XAIRec that integrates KGs and RL. The framework of XAIRec utilizes three algorithms. The first primary algorithm is RUPPEP, which highlights the effectiveness of incorporating the learning from historical purchases. The RL model developed through the RUPEP algorithm assists in predicting the crucial parameters of the number of rules/features/hops ($\mathbb{R}$),

probability of the traversal path ($\mathbb{P}$), entropy/information value ($\mathbb{H}$), affinity score ($\mathbb{S}$), and rewards ($\mathbb{Rw}$) for the potential recommended products. During the recommendation phase, the proposed framework applies MES and PPS algorithms in improving product RSs. The results showed significant performance improvements in critical evaluation metrics such as NDCG, HR, Recall, and Precision, emphasizing the positive influence of the XAIRec on the overall performance of the recommendation engine. The combinations of RUPPEP, MES, and PPS enhanced the accuracy of product recommendations and provided explainability to users, which is essential for user trust and satisfaction.

The research has addressed a research gap by introducing a method to quantify the explainability measures of knowledge-graph-based explanations. By incorporating the parameters of the number of rules/features/hops ($\mathbb{R}$), probability of the traversal path ($\mathbb{P}$), entropy/information value ($\mathbb{H}$), and rewards ($\mathbb{Rw}$), this study has proposed the MES or $\boldsymbol{\mathcal{H}}^*$ metric. This metric effectively combines rewards gain ($\mathbb{Rw}_{Gain}$), entropy gain ($\mathbb{H}_{Gain}$), and the number of features ($\mathbb{R}$) in the user-traversed path, providing a comprehensive measure of explainability.

The experimental results highlight the effectiveness of the MES or $\boldsymbol{\mathcal{H}}^*$ metric in evaluating the recommendation model. The utilisation of MES or $\boldsymbol{\mathcal{H}}^*$ has shown improvements in crucial evaluation metrics such as NDCG, HR, Recall, and Precision, emphasising the positive influence of high explainability on the overall performance of the recommendation engine. The utilisation of the optimal path selection approach based on explainability scores has not only enhanced the accuracy of the recommendations but also provided users with meaningful and interpretable explanations for the recommended products.

The PPS algorithm makes a significant contribution to the XAIRec framework by effectively prioritizing products based on user priorities. This prioritization enhances the relevance and personalization of recommendations, leading to improved user satisfaction. By integrating the PPS algorithm, the XAIRec framework can better balance exploration and exploitation, ensuring that the recommended products not only align with user preferences but also optimize the overall recommendation

performance. The PPS algorithm's ability to dynamically adjust priorities based on user behaviour and feedback makes it a crucial component in delivering high-quality, context-aware recommendations.

This research has made significant contributions by introducing a method for quantifying the explainability measures of generated explanations and inducing transparency and interpretability into RSs. These advancements have led to several benefits, including enhanced user trust, reduced bias, and improved overall user experience. With a better understanding of the rationale behind specific recommendations, users can make more informed judgments and have increased confidence in the system's decision-making process. As a result, users develop a stronger sense of trust and satisfaction with the recommendations provided. This improved understanding and confidence ultimately enable the RS to deliver a more personalised and user-centric experience.

By continually improving the transparency and interpretability of RSs, this study can pave the way for more trustworthy, personalised, and user-centric recommendations in various domains, benefiting both users and service providers alike.

To achieve the main research goal of this study, some sub-research questions, objectives, and problems related to this study's main problem are addressed and shown in **Table 7.1**.

Table 7.1 Connection of Research Objectives and Study's Achievements

Research Objectives	Achievement of the Research Objectives	Covered in Section
RO1: To propose a framework that mitigates the limitations (**not rewarding agents for pursuing intermediate paths that are subsets of guided path patterns and may align with user preferences captured based on their historical purchases**) found in the current state-of-the-art baseline models by extending the proposed technique as a component of the intended framework approach.	This research proposed a novel algorithm named RUPPEP in section 4.4.1. This algorithm departs from the conventional RL approach by providing rewards to the RL agent at each step of its traversal path, as opposed to solely at the final step. This intermediary reward mechanism enables the agent to learn and adapt its actions more effectively to maximize the cumulative rewards. The agent receives the highest rewards when it successfully reaches the previously purchased targeted product on behalf of the user. In essence, this approach incentivizes the agent to continuously gauge and respond to user preferences throughout the decision-making process. A thorough analysis conducted in Chapter 6 illustrates the efficacy of the proposed XAIRec framework.	Chapter 6 & Section 4.4.1 – Algorithm 1
RO2: To enhance the **embedded explanation** of the RSs by **identifying the optimal explainability path**, which not only provides better explainability but also assists in improving the performance of the recommendations.	This research addresses the challenge of determining the most crucial connectivity path for recommending products to the user when there are multiple possible options. It proposed a novel algorithm, the Max Explainability Score in section 4.4.3, which assists in identifying the most valuable path that should take precedence in decision-making. The significance of this preferred path extends beyond its ability to enhance explainability. It also plays a pivotal role in improving the overall decision-making process for product recommendations. By selecting and prioritizing this path, the algorithm provides the reasoning or explanation behind the product recommendation and contributes to more effective and well-informed product recommendations.	Section 6.2 & Section 4.4.3 – Algorithm 2
RO3: To enhance the **semantic recommendation** of the RSs by **prioritizing the products useful for the user** and improving the recommendation performances.	After determining the optimal connectivity path for candidate products, the subsequent challenge is the top-N product recommendation. In this context, numerous products may pique the user's interest, but the key question is how to rank and prioritize these products based on their compatibility with the user's preferences. In addressing this issue, the research introduces an advanced algorithm called the "Products Prioritisation Score," as detailed in section 4.4.4. This algorithm plays a pivotal role in aiding the prioritization of products most aligned with the user's preferences. Chapter 6 and section 6.3 of the research showcase the overall effectiveness of the XAIRec framework, primarily attributable to the adoption of the PPS algorithm.	Section 6.3 & Section 4.4.4 – Algorithm 3

to be continued…

… continuation

RO4: To propose a **quantitative measure to evaluate the explainability efficacies**, a current research gap in quantitatively measuring the explainability efficacies of generated explainabilities of the provided recommendations by the XRSs.	The study identified a research gap and sought to address it by proposing a novel algorithm in section 4.5.1 for quantitatively assessing the effectiveness of generated explanations. It assigns an explainability score to the generated explanations and quantifies their importance for the user. The effectiveness of this approach is demonstrated in Section 6.4, where the study illustrates a correlation between the generated explainabilities and product recommendations. In other words, the study shows that explanations with higher explainability scores are associated with more effective product recommendations, providing valuable insights into the importance of explainability in the recommendation process.	Section 6.4 & Section 4.5.1 – Algorithm 4

7.3 CONTRIBUTION OF THE RESEARCH

7.3.1 Explainable Recommendations

This study presents a novel approach called XAIRec to enhance accuracy, transparency, and interpretability in KG-based RSs. It combines three advanced algorithms: RUPPEP, which uses path embedding propagation to drive user preferences by rewarding agents for exploring intermediate paths that are subsets of guided path patterns aligned with user preferences from historical purchases; MES, which determines the optimal traversal path considering multiple factors to provide optimal explainability and assist in product prioritization using PPS methods. The RUPPEP method utilizes KGs to generate probabilities, rewards, and other metrics for potential traversal paths of recommended products, facilitating a better understanding of user preferences. The MES method plays a crucial role in identifying the optimal traversal path within the KG for the generated recommendations and thus enhances interpretability and explainability. Additionally, the PPS method ranks products to prioritize those that align with user interests, improving the quality of the top-N recommendations.

The contributions of this research can be summarised as follows.

a. It proposes an RL-based approach driven by the RUPPEP component to leverage historical purchasing information, incorporate a soft reward strategy, user-conditional action pruning, and a multi-hop scoring strategy to learn users' preferences. This approach enhances the model's ability to understand user behaviour patterns and make relevant recommendations.

b. It introduces a novel MES algorithm to efficiently determine optimal reasoning paths and candidate item sets, resulting in more interpretable recommendations and assisting in evaluating the explainabilities.

c. A product prioritization algorithm PPS assists in further improving the quality of the recommendations by prioritizing relevant products during the recommendation phase.

Extensive evaluations across various Amazon e-commerce domains validate the strong performance of the proposed framework, showcasing its ability to provide explainable reasoning paths for the recommended items. While the RUPPEP component leverages historical purchases to guide the agent's learning of a policy for future recommendations, the MES component enhances explainability by providing insights into the underlying model's decision-making process and generating a quantitative measure of explainability. PPS component prioritizes the products for the user, ensuring trustworthiness and performance. Overall, XAIRec significantly contributes to ongoing efforts in developing accurate and XRS to guide the recommendation process, ensure explainability, and enhance user satisfaction.

7.3.2 Explainable Recommendations – Quantitative Evaluation

It proposes a new quantitative metric MES, to measure recommendation explainability in the framework of KGs, RL, and other related technologies. I believe this research is the first toward quantifying explainability metrics in the recommendation engines with the KG.

The goal of explainability is to justify any recommendations offered to users to increase their confidence in the system. At the core of quality explainability is the

surprise information the explainability possesses. The surprise or valuable information the explanation model fetches depends on a few factors, such as the number of attributable features, the quality of those attributable features (i.e., the probability of occurrence of those attributable features in the path of generating recommendations), the information value that these attributable features hold (i.e., the entropy due to the opted traversal paths), the affinity of the recommended product for the user, and the rewards that the traversal path gets for opting the specific route.

This research proposes a new explainability measuring metric MES (or $\mathcal{H}^*$) based on the suggested four evaluation parameters: $\mathbb{R}$ - the number of rules/features/hops outputted by the explanation, $\mathbb{P}$ - the probability of the traversal path for the recommended item, $\mathbb{H}$ – the entropy/information value due to the opted traversal path for the recommended item, and $\mathbb{Rw}$ - the reward due to the opted traversal paths for the recommended items. This study believes that user studies focusing on these metrics in their evaluations are inherently more valid and may better integrate with future XAI research.

7.4 FUTURE WORKS SUGGESTIONS

The findings of this study carry significant implications for the field of RSs, emphasising the significance of integrating KGs and explainability metrics. These findings pave the way for future research endeavours to develop more transparent and interpretable recommendation models.

Further investigations should focus on assessing the scalability and efficiency of the proposed framework when applied to larger KGs and real-world scenarios. Additionally, conducting user studies to evaluate the perceived usefulness and comprehensibility of the explanations provided would yield deeper insights into the practical implications of the approach. By addressing these areas, this study can advance the field and foster the adoption of more effective and user-friendly RSs.

Also, further research can explore the integration of additional algorithms and techniques to enhance the explainability and interpretability of the recommendation

process. This could involve incorporating more sophisticated machine learning models, advanced natural language processing techniques, or leveraging user feedback and preferences for personalized explanations.

Additionally, investigating the impact of different parameter settings and variations in the RUPPEP, MES, and PPS algorithms could provide insights into optimizing the recommendation process. Fine-tuning these parameters, while developing the KGE and RL models, and exploring their effects on performance metrics can lead to improved recommendations and tailored explanations for individual users.

Furthermore, conducting user studies and gathering feedback on the generated explanations can help refine the proposed framework and ensure that the generated explanations are meaningful, understandable, and meet user expectations. User feedback can also shed light on specific preferences and requirements related to explainability, allowing for the development of user-centric RSs.

In conclusion, while the experiment demonstrated the effectiveness of RUPPEP, MES, and PPS algorithms in enhancing product recommendation and providing explainability, future work can focus on further improving the methodology, exploring additional algorithms and techniques, optimizing parameter settings, and incorporating user feedback to create more robust and personalized RSs.

7.5 SUMMARY

The comprehensive conclusion of this chapter synthesizes the primary insights garnered from both the research findings in Section 7.2 and the prospective directions outlined in Section 7.4. It underscores the significance of this research's impact on the field of XAI and RSs. By enhancing the transparency of RSs and offering actionable pathways for future work, this study contributes to a growing body of knowledge aimed at improving the interpretability and trustworthiness of AI-driven recommendations. In doing so, it not only addresses current challenges but also advances the broader vision of XAI, promoting more effective and responsible AI applications in various domains.

REFERENCES

Abdollahi, B., & Nasraoui, O. (2017). Using Explainability for Constrained Matrix Factorization. *In Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)* (pp. 79-83). New York, NY, United States: Association for Computing Machinery. https://doi.org/10.1145/3109859.3109913

Aciar, S., Zhang, D., Simoff, S., & Debenham, J. (2007). Informed Recommender: Basing Recommendations on Consumer Product Reviews. *IEEE Intelligent Systems, 22*(3), 39-47. https://doi.org/10.1109/MIS.2007.55

Adadi, A., & Berrada, M. (2018). Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access, 6*, 52138-52160. https://doi.org/10.1109/ACCESS.2018.2870052

Adomavicius, G., Manouselis, N., & Kwon, Y. (2011). Multi-Criteria Recommender Systems. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook* (pp. 769-803). Boston, MA: Springer US. https://doi.org/10.1007/978-0-387-85820-3_24

Afsar, M. M., Crump, T., & Far, B. (2022). Reinforcement Learning based Recommender Systems: A Survey. *ACM Computing Surveys, 55*(7), 1-38. https://doi.org/10.1145/3543846

Ai, Q., Azizi, V., Chen, X., & Zhang, Y. (2018). Learning Heterogeneous Knowledge Base Embeddings for Explainable Recommendation. *Algorithms, 11*(9), 137. https://doi.org/10.3390/a11090137

Alamdari, P. M., Navimipour, N. J., Hosseinzadeh, M., Safaei, A. A., & Darwesh, A. (2020). A Systematic Study on the Recommender Systems in the E-Commerce. *IEEE Access, 8*, 115694-115716. https://doi.org/10.1109/ACCESS.2020.3002803

Al-Ghuribi, S. M., & Noah., S. A. (2021). Study on Various Collabrative Filtering Techniques to Recommend Movies. *Multi-criteria review-based recommender system–the state of the art, 7*, 169446-169468.

Ali, S., Abuhmed, T., El-Sappagh, S., Muhammad, K., Alonso-Moral, J. M., Confalonieri, R., Guidotti, R., Ser, J. D., Díaz-Rodríguez, N., & Herrera, F. (2023). Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. *Information Fusion, 99*, 101805. https://doi.org/https://doi.org/10.1016/j.inffus.2023.101805

Amatriain, X., & Basilico, J. (2015). Recommender systems in industry: A netflix case study. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (2 ed., pp. 385--419). New York, NY: Springer. https://doi.org/10.1007/978-1-4899-7637-6

Arrieta, A. B., Díaz-Rodríguez, N., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion, 58*, 82-115. https://doi.org/https://doi.org/10.1016/j.inffus.2019.12.012

Benjamins, R., Barbado, A., & Sierra, D. (2019). Responsible AI by design in practice. *arXiv preprint arXiv:1909.12838*. https://doi.org/https://doi.org/10.48550/arXiv.1909.12838

Björklund, G., Bohlin, M., Olander, E., Jansson, J., Walter, C., & Au-Yong-Oliveira, M. (2022). An Exploratory Study on the Spotify Recommender System. In A. Rocha, H. Adeli, G. Dzemyda, & F. Moreira (Eds.), *World Conference on Information Systems and Technologies. WorldCIST 2022. Lecture Notes in Networks and Systems* (Vol. 469, pp. 366--378). Springer, Cham. https://doi.org/10.1007/978-3-031-04819-7_36

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *In Proceedings of the 2008 ACM SIGMOD international conference on Management of data (SIGMOD '08)* (pp. 1247-1250). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1376616.1376746

Bonatti, P. A., Decker, S., Polleres, A., & Presutti, V. (2019). Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371). (P. A. Bonatti, S. Decker, A. Polleres, & V. Presutti, Eds.) *Dagstuhl Reports, 8*(9), 29-111. https://doi.org/10.4230/DagRep.8.9.29

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating Embeddings for Modeling Multi-relational Data. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 26). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

Bunn, J. (2020). Working in contexts for which transparency is important: A recordkeeping view of explainable artificial intelligence (XAI). *Records Management Journal, 30*(2), 143--153. https://doi.org/10.1108/RMJ-08-2019-0038

Cai, H., Zheng, V. W., & Chang, K. C.-C. (2018). A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE transactions on knowledge and data engineering, 30*(9), 1616-1637. https://doi.org/10.1109/TKDE.2018.2807452

Cano, E., & Morisio, M. (2017). Hybrid Recommender Systems: A Systematic Literature Review. *Intelligent Data Analysis, 21*(6), 1487-1524. https://doi.org/10.3233/IDA-163209

Cao, J., Fang, J., Meng, Z., & Liang, S. (2022). Knowledge Graph Embedding: A Survey from the Perspective of Representation Spaces. *arXiv preprint arXiv:2211.03536*. https://doi.org/10.48550/arXiv.2211.03536

Cao, Y., Wang, X., He, X., Hu, Z., & Chua, T.-S. (2019). Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *In The World Wide Web Conference (WWW '19)* (pp. 151-161). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3308558.3313705

Chen, G., & Chen, L. (2014). Recommendation Based on Contextual Opinions. In V. Dimitrova, T. Kuflik, D. Chin, F. Ricci, P. Dolog, & G. Houben (Eds.), *User Modeling, Adaptation, and Personalization: 22nd International Conference, UMAP 2014, Aalborg, Denmark, July 7-11, 2014. Proceedings 22* (Vol. 8538, pp. 61-73). Aalborg, Denmark,: Springer, Cham. https://doi.org/10.1007/978-3-319-08786-3_6

Chen, L., Chen, G., & Wang, F. (2015). Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction, 25*(2), 99-154. https://doi.org/10.1007/s11257-015-9155-5

Chen, R., Hua, Q., Chang, Y.-S., Wang, B., Zhang, L., & Kong, X. (2018). A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks. *IEEE Access, 6*, 64301-64320. https://doi.org/10.1109/ACCESS.2018.2877208

Chen, X., Yao, L., McAuley, J., Zhou, G., & Wang, X. (2023). Deep reinforcement learning in recommender systems: A survey and new perspectives. *Knowledge-Based Systems, 264*, 110335. https://doi.org/https://doi.org/10.1016/j.knosys.2023.110335

Chopra, R., & Roy, S. S. (2020). End-to-End Reinforcement Learning for Self-driving Car. In B. Pati, C. R. Panigrahi, R. Buyya, & K.-C. Li (Eds.), *Advanced Computing and Intelligent Engineering: Proceedings of ICACIE 2018* (Vol. 1082, pp. 53-61). Singapore: Springer, Singapore. https://doi.org/10.1007/978-981-15-1081-6_5

Chu, W.-T., & Tsai, Y.-L. (2017). A hybrid recommendation system considering visual information for predicting favorite restaurants. *World Wide Web, 20*(6), 1313--1331. https://doi.org/10.1007/s11280-017-0437-1

Clancey, W. J. (1983). The epistemology of a rule-based expert system—a framework for explanation. *Artificial Intelligence, 20*(3), 215-251. https://doi.org/https://doi.org/10.1016/0004-3702(83)90008-5

Cui, H., Peng, T., Xiao, F., Han, J., Han, R., & Liu, L. (2023). Incorporating anticipation embedding into reinforcement learning framework for multi-hop knowledge graph question answering. *Information Sciences, 619*, 745-761. https://doi.org/https://doi.org/10.1016/j.ins.2022.11.042

D'Addio, R. M., & Manzato, M. G. (2014). A Collaborative Filtering Approach Based on User's Reviews. In *2014 Brazilian Conference on Intelligent Systems* (pp. 204--209). IEEE. https://doi.org/10.1109/BRACIS.2014.45

Daneshvar, H., & Ravanmehr, R. (2022). A social hybrid recommendation system using LSTM and CNN. *Concurrency and Computation: Practice and Experience, 34*(18), e7015. https://doi.org/https://doi.org/10.1002/cpe.7015

Das, R., Dhuliawala, S., Zaheer, M., Vilnis, L., Durugkar, I., Krishnamurthy, A., Smola, A., & McCallum, A. (2017). Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases with Reinforcement Learning. *NIPS-17 Workshop on Automatic Knowledge-Base Construction*. https://doi.org/https://doi.org/10.48550/arXiv.1711.05851

de la Torre, L. (2018, November). A Guide to the California Consumer Privacy Act of 2018. *Available at SSRN 3275571*. https://doi.org/https://dx.doi.org/10.2139/ssrn.3275571

Deeks, A. (2019). THE JUDICIAL DEMAND FOR EXPLAINABLE ARTIFICIAL INTELLIGENCE. *Columbia Law Review, 119*(7), 1829--1850. https://www.jstor.org/stable/26810851

Dias, R., & Fonseca, M. J. (2013). Improving Music Recommendation in Session-Based Collaborative Filtering by Using Temporal Context. In *2013 IEEE 25th International Conference on Tools with Artificial Intelligence* (pp. 783-788). Herndon, VA, USA: IEEE. https://doi.org/10.1109/ICTAI.2013.120

Douban. (2005). *Douban movie*. http://movie.douban.com/

Ebadi, A., & Krzyzak, A. (2016). A Hybrid Multi-Criteria Hotel Recommender System Using Explicit and Implicit Feedbacks. *International Journal of Computer and Information Engineering, 10*(8), 1450-1458. https://doi.org/https://doi.org/10.5281/zenodo.1125867

Ehsan, U., Liao, Q. V., Muller, M., Riedl, M. O., & Weisz, J. D. (2021). Expanding Explainability: Towards Social Transparency in AI Systems. In *In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)* (pp. 1-19). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3411764.3445188

Evans, T., Retzlaff, C. O., Geißler, C., Kargl, M., Plass, M., Müller, H., Kiehl, T.-R., Zerbe, N., & Holzinger, A. (2022). The explainability paradox: Challenges for xAI in digital pathology. *Future Generation Computer Systems, 133*, 281-296. https://doi.org/https://doi.org/10.1016/j.future.2022.03.009

Fang, Y., Lin, W., Zheng, V. W., Wu, M., Chang, K. C.-C., & Li, X.-L. (2016). Semantic proximity search on graphs with metagraph-based learning. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)* (pp. 277-288). Helsinki, Finland: IEEE. https://doi.org/10.1109/ICDE.2016.7498247

Fayyaz, Z., Ebrahimian, M., Nawara, D., Ibrahim, A., & Kashef, R. (2020). Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities. *Applied Sciences, 10*(21), 7748. https://www.mdpi.com/2076-3417/10/21/7748

Fensel, D., Şimşek, U., Angele, K., Huaman, E., Kärle, E., Panasiuk, O., Toma, I., Umbrich, J., & Wahler, A. (2020). *Knowledge Graphs* (1 ed.). Switzerland: Springer Cham. https://doi.org/https://doi.org/10.1007/978-3-030-37439-6

Gao, L., Yang, H., Wu, J., Zhou, C., Lu, W., & Hu, Y. (2018). Recommendation with multi-source heterogeneous information. *IJCAI International Joint Conference on Artificial Intelligence* (pp. 3378-3384). OPUS at UTS. http://hdl.handle.net/10453/131480

García-Cumbreras, M. Á., Montejo-Ráez, A., & Díaz-Galiano, M. C. (2013). Pessimists and optimists: Improving collaborative filtering through sentiment analysis. *Expert Systems with Applications, 40*(17), 6758-6765. https://doi.org/https://doi.org/10.1016/j.eswa.2013.06.049

Gedikli, F., Jannach, D., & Ge, M. (2014). How should I explain? A comparison of different explanation types for recommender systems. *International Journal of Human-Computer Studies, 72*(4), 367-382. https://doi.org/https://doi.org/10.1016/j.ijhcs.2013.12.007

Geng, S., Fu, Z., Tan, J., Ge, Y., De Melo, G., & Zhang, Y. (2022). Path Language Modeling over Knowledge Graphsfor Explainable Recommendation. *In Proceedings of the ACM Web Conference 2022 (WWW '22)* (pp. 946-955). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3485447.3511937

Gerlings, J., Jensen, M. S., & Shollo, A. (2022). Explainable AI, But Explainable to Whom? An Exploratory Case Study of xAI in Healthcare. In C.-P. Lim, Y.-W. Chen, A. Vaidya, C. Mahorkar, & L. C. Jain (Eds.), *Handbook of Artificial Intelligence in Healthcare: Vol 2: Practicalities and Prospects* (Vol. 212, pp. 169--198). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-83620-7_7

Gigerenzer, G., & Brighton, H. (2009). Homo Heuristicus: Why Biased Minds Make Better Inferences. *Topics in cognitive science, 1*(1), 107-143. https://doi.org/10.1111/j.1756-8765.2008.01006.x

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)* (pp. 80--89). IEEE.

GroupLens. (1997). *Movielens dataset*. https://grouplens.org/datasets/movielens/

GroupLens. (2011). *Last.fm-A dataset*. https://grouplens.org/datasets/hetrec-2011/

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable Feature Learning for Networks. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)* (pp. 855-864). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2939672.2939754

Guan, X., Li, C.-T., & Guan, Y. (2017). Matrix Factorization With Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems. *IEEE Access, 5*, 27668-27678. https://doi.org/10.1109/ACCESS.2017.2772226

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A Survey of Methods for Explaining Black Box Models. *ACM computing surveys (CSUR), 51*(5), 1-42. https://doi.org/10.1145/3236009

Gunawardana, A., Shani, G., & Yogev, S. (2012). Evaluating Recommender Systems. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 547-601). New York, NY: Springer US. https://doi.org/10.1007/978-1-0716-2197-4_15

Gunning, D., & Aha, D. (2019). DARPA's Explainable Artificial Intelligence (XAI) Program. *AI magazine, 40*(2), 44-58. https://doi.org/10.1609/aimag.v40i2.2850

Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., & Yang, G.-Z. (2019). XAI—Explainable artificial intelligence. *Science Robotics, 4*(37), eaay7120. https://doi.org/10.1126/scirobotics.aay7120

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2020, August). A Survey on Knowledge Graph-Based Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering, 34*(8), 3549-3568. https://doi.org/10.1109/TKDE.2020.3028705

Hailemariam, Y., Yazdinejad, A., Parizi, R. M., Srivastava, G., & Dehghantanha, A. (2020). An empirical evaluation of AI deep explainable tools. In *2020 IEEE Globecom Workshops (GC Wkshps* (pp. 1--6). Taipei, Taiwan: IEEE. https://doi.org/10.1109/GCWkshps50303.2020.9367541

Han, J., Zheng, L., Xu, Y., Zhang, B., Zhuang, F., Philip, S. Y., & Zuo, W. (2020). Adaptive Deep Modeling of Users and Items Using Side Information for Recommendation. *IEEE Transactions on Neural Networks and Learning Systems, 31*(3), 737-748. https://doi.org/10.1109/TNNLS.2019.2909432

Hao, J., Chen, M., Yu, W., Sun, Y., & Wang, W. (2019). Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)* (pp. 1709–1719). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3292500.3330838

He, R., & McAuley, J. (2016). Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. *In Proceedings of the*

25th International Conference on World Wide Web (WWW '16) (pp. 507-517). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/2872427.2883037

He, R., Kang, W.-C., & McAuley, J. (2017). Translation-based Recommendation. *In Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)* (pp. 161-169). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3109859.3109882

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems (TOIS), 22*(1), 5--53. https://doi.org/10.1145/963770.963772

Heuillet, A., Couthouis, F., & Diaz-Rodriguez, N. (2021). Explainability in deep reinforcement learning. *Knowledg- Based Systems, 214*, 106685. https://doi.org/https://doi.org/10.1016/j.knosys.2020.106685

Hoffman, R. R., Mueller, S. T., Klein, G., & Litman, J. (2018). Metrics for explainable AI: Challenges and prospects. *arXiv preprint arXiv:1812.04608*. https://doi.org/https://doi.org/10.48550/arXiv.1812.04608

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E., Navigli, R., Neumaier, S., & others. (2021). Knowledge Graphs. *ACM Computing Surveys (Csur), 54*(4), 1-37. https://doi.org/10.1145/3447772

Holzinger, A., Langs, G., Denk, H., Zatloukal, K., & Müller, H. (2019). Causability and explainability of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9*(4), e1312. https://doi.org/10.1002/widm.1312

Hu, B., Shi, C., Zhao, W. X., & Yu, P. S. (2018). Leveraging Meta-path based Context for Top- N Recommendation with A Neural Co-Attention Model. *In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)* (pp. 1531-1540). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3219819.3219965

Huang, J., Zhao, W. X., Dou, H., Wen, J.-R., & Chang, E. Y. (2018). Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. *In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)* (pp. 505--514). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3209978.3210017

Huang, X., Fang, Q., Qian, S., Sang, J., Li, Y., & Xu, C. (2019). Explainable Interaction-driven User Modeling over Knowledge Graph for Sequential Recommendation. *In Proceedings of the 27th ACM International Conference*

on Multimedia (MM '19) (pp. 548--556). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3343031.3350893

Huang, Z., Liu, Q., Zhai, C., Yin, Y., Chen, E., Gao, W., & Hu, G. (2019). Exploring Multi-Objective Exercise Recommendations in Online Education Systems. In *In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)* (pp. 1261-1270). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3357384.3357995

Ji, G., He, S., Xu, L., Liu, K., & Zhao, J. (2015). Knowledge Graph Embedding via Dynamic Mapping Matrix. In *roceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 687-696). Beijing, China: Association for Computational Linguistics. https://doi.org/10.3115/v1/p15-1067

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2022). A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Transactions on Neural Networks and Learning Systems, 33*(2), 494-514. https://doi.org/10.1109/TNNLS.2021.3070843

Khalil, A., Elmogy, M., Ghazal, M., Burns, C., & El-Baz, A. (2019). Chronic Wound Healing Assessment System Based on Different Features Modalities and Non-Negative Matrix Factorization (NMF) Feature Reduction. *IEEE Access, 7*, 80110--80121. https://doi.org/10.1109/ACCESS.2019.2923962

Kim, K. H., Ko, E., Kim, S. J., & Jiang, Q. (2021). Digital service innovation, customer engagement, and customer equity in AR marketing. *Journal of Global Scholars of Marketing Science, 31*(3), 453--466. https://doi.org/10.1080/21639159.2021.1923054

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., & P{\'e}rez, P. (2021). Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems, 23*, 4909--4926. https://doi.org/http://dx.doi.org/10.1109/TITS.2021.3054625

KKBox. (2018). *Kkbox dataset*. https://wsdm-cup-2018.kkbox.events/

Klein, N., Ilievski, F., Freedman, H., & Szekely, P. (2022). Identifying Surprising Facts in Wikidata. *Wikidata '22*.

Konda, V., & Tsitsiklis, J. (1999). Actor-Critic Algorithms. *Advances in Neural Information Processing Systems, 12*. https://proceedings.neurips.cc/paper_files/paper/1999/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer, 42*(8), 30--37. https://doi.org/10.1109/MC.2009.263

Kotkov, D., Veijalainen, J., & Wang, S. (2017). A Serendipity-Oriented Greedy Algorithm for Recommendations. In *Proceedings of the 13th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST* (pp. 32-40). SCITEPRESS Science And Technology Publications. https://doi.org/10.5220/0006232800320040

Kotriwala, A., Klöpper, B., Dix, M., Gopalakrishnan, G., Ziobro, D., & Potschka, A. (2021). XAI for Operations in the Process Industry-Applications, Theses, and Research Directions. *Proceedings of the AAAI 2021 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021) - Stanford University* (pp. 1-12). Palo Alto, California, USA: AAAI. https://proceedings.aaai-make.info/AAAI-MAKE-PROCEEDINGS-2021/paper26.pdf

Last.fm. (2002). *Last.fm online music system*. http://www.last.fm/

Laux, J., Wachter, S., & Mittelstadt, B. (2023). Trustworthy artificial intelligence and the European Union AI act: On the conflation of trustworthiness and acceptability of risk. *Regulation & Governance*. https://doi.org/10.1111/rego.12512

Lawrence, R., Almasi, G., Kotlyar, V., Viveros, M., & Duri, S. (2001). Personalization of Supermarket Product Recommendations. In *Applications of Data Mining to Electronic Commerce* (pp. 11--32). Boston, MA: Springer US. https://doi.org/10.1007/978-1-4615-1627-9_2

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., & others. (2015). DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic web*, *6*(2), 167-195. https://doi.org/10.3233/SW-140134

Li, Q., Tang, X., Wang, T., Yang, H., & Song, H. (2019). Unifying Task-Oriented Knowledge Graph Learning and Recommendation. *IEEE Access*, *7*, 115816--115828. https://doi.org/10.1109/ACCESS.2019.2932466

Li, X., Li, X., Pan, D., & Zhu, D. (2020). On the Learning Property of Logistic and Softmax Losses for Deep Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*, pp. 4739-4746. https://doi.org/10.1609/aaai.v34i04.5907

Lim, B. Y., Dey, A. K., & Avrahami, D. (2009). Why and why not explanations improve the intelligibility of context-aware intelligent systems. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)* (pp. 2119--2128). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1518701.1519023

Lin, Y., Liu, Z., Sun, M., Liu, Y., & Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 29).

Lin, Y.-S., Wen-Chuan, L., & Celik, Z. B. (2021). What Do You See? Evaluation of Explainable Artificial Intelligence (XAI) Interpretability through Neural Backdoors. *In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '21)* (pp. 1027--1035). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3447548.3467213

Liu, H., Wang, Y., Peng, Q., Wu, F., Gan, L., Pan, L., & Jiao, P. (2020). Hybrid neural recommendation with joint deep representation learning of ratings and reviews. *Neurocomputing, 374*, 77-85. https://doi.org/https://doi.org/10.1016/j.neucom.2019.09.052

Liu, N., & Zhao, J. (2023). Recommendation System Based on Deep Sentiment Analysis and Matrix Factorization. *IEEE Access, 11*, 16994-17001. https://doi.org/10.1109/ACCESS.2023.3246060

Liu, P., Zhang, L., & Gulla, J. A. (2020). Dynamic attention-based explainable recommendation with textual and visual fusion. *Information Processing & Management*, *57*(6), 102099. https://doi.org/https://doi.org/10.1016/j.ipm.2019.102099

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: a survey. *Decision Support Systems, 74*, 12-32. https://doi.org/https://doi.org/10.1016/j.dss.2015.03.008

Lu, Y., Dong, R., & Smyth, B. (2018). Coevolutionary Recommendation Model: Mutual Learning between Ratings and Reviews. *In Proceedings of the 2018 World Wide Web Conference (WWW '18)* (pp. 773--782). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/3178876.3186158

Luo, C., Pang, W., Wang, Z., & Lin, C. (2014). Hete-CF: Social-Based Collaborative Filtering Recommendation Using Heterogeneous Relations. In *2014 IEEE International Conference on Data Mining* (pp. 917-922). Shenzhen, China: IEEE. https://doi.org/10.1109/ICDM.2014.64

Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., & Ren, X. (2019). Jointly Learning Explainable Rules for Recommendation with Knowledge Graph. *In The World Wide Web Conference (WWW '19)* (pp. 1210-1221). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3308558.3313607

Malleswari, N. V., Gayatri, K., Kumar, K. Y., Likhita, N., Padmanaban, K., & Bhattacharyya, D. (2023). Music Recommendation System using Hybrid Approach. In *2023 Second International Conference on Electronics and Renewable Systems (ICEARS)* (pp. 1560-1564). Tuticorin, India: IEEE. https://doi.org/10.1109/ICEARS56392.2023.10085059

Mankodiya, H., Obaidat, M. S., Gupta, R., & Tanwar, S. (2021). XAI-AV: Explainable Artificial Intelligence for Trust Management in Autonomous

Vehicles. *2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)* (pp. 1-5). Beijing, China: IEEE. https://doi.org/10.1109/CCCI52664.2021.9583190

Marcuzzo, M., Zangari, A., Albarelli, A., & Gasparetto, A. (2022). Recommendation Systems: An Insight Into Current Development and Future Research Challenges. *IEEE Access, 10*, 86578-86623. https://doi.org/10.1109/ACCESS.2022.3194536

McAuley, J., & Leskovec, J. (2013). Hidden factors and hidden topics: understanding rating dimensions with review text. *In Proceedings of the 7th ACM conference on Recommender systems (RecSys '13)* (pp. 165--172). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2507157.2507163

McAuley, J., Targett, C., Shi, Q., & Van Den Hengel, A. (2015). Image-Based Recommendations on Styles and Substitutes. *In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)* (pp. 43-52). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2766462.2767755

Mohseni, S., Zarei, N., & Ragan, E. D. (2021). A Multidisciplinary Survey and Framework for Design and Evaluation of Explainable AI Systems. *ACM Transactions on Interactive Intelligent Systems (TiiS), 11*(3-4), 1-45. https://doi.org/10.1145/3387166

MovieLens. (1997). *Movielens website*. https://movielens.org/

Nawara, D., & Kashef, R. (2021). Deploying Different Clustering Techniques on a Collaborative-based Movie Recommender. *2021 IEEE International Systems Conference (SysCon)* (pp. 1-6). Vancouver, BC, Canada: IEEE. https://doi.org/10.1109/SysCon48628.2021.9447139

O'Dair, M., & Fry, A. (2020). Beyond the black box in music streaming: the impact of recommendation systems upon artists. *Popular Communication, 18*(1), 65-77. https://doi.org/10.1080/15405702.2019.1627548

Ohana, J. J., Ohana, S., Benhamou, E., Saltiel, D., & Guez, B. (2021). Explainable AI (XAI) Models Applied to the Multi-agent Environment of Financial Markets. In *Explainable and Transparent AI and Multi-Agent Systems: Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3--7, 2021, Revised Selected Papers 3* (Vol. 12688, pp. 189-207). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-82017-6_12

Palumbo, E., Rizzo, G., & Troncy, R. (2017). Entity2rec: Learning User-Item Relatedness from Knowledge Graphs for Top-N Item Recommendation. *In Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)* (pp. 32-36). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3109859.3109889

Pan, Y., He, F., & Yu, H. (2020). Learning social representations with deep autoencoder for recommender system. *World Wide Web, 23*(4), 2259-2279. https://doi.org/10.1007/s11280-020-00793-z

Pazzani, M. J., & Billsus, D. (2007). Content-Based Recommendation Systems. In *The Adaptive Web: Methods and Strategies of Web Personalization* (pp. 325-341). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-540-72079-9_10

Peake, G., & Wang, J. (2018). Explanation Mining: Post Hoc Interpretability of Latent Factor Models for Recommendation Systems. *In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)* (pp. 2060-2069). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3219819.3220072

Perdih, T. S., Lavrač, N., & Škrlj, B. (2021). Semantic Reasoning from Model-Agnostic Explanations. In *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)* (pp. 000105-000110). Herl'any, Slovakia: IEEE. https://doi.org/10.1109/SAMI50585.2021.9378668

Phillips-Wren, G., & Adya, M. (2020). Decision making under stress: The role of information overload, time pressure, complexity, and uncertainty. *Journal of Decision Systems, 29*(sup1), 213-225. https://doi.org/10.1080/12460125.2020.1768680

Plaat, A., Kosters, W., & Preuss, M. (2023). High-accuracy model-based reinforcement learning, a survey. *Artificial Intelligence Review, 56*(9), 9541-9573. https://doi.org/10.1007/s10462-022-10335-w

Purohit, S., Van, N., & Chin, G. (2021). Semantic Property Graph for Scalable Knowledge Graph Analytics. *2021 IEEE International Conference on Big Data (Big Data)* (pp. 2672-2677). Orlando, FL, USA: IEEE. https://doi.org/10.1109/BigData52589.2021.9671547

R. Wickman. (2022). SPARRL: Graph sparsification via deep reinforcement. In *Proc. ACM SIGMOD Int. Conf. Manage.Data, Philadelphia,* (pp. 2521–2523). Philadelphia, USA: ACM.

Rachha, A., & Seyam, M. (2023). Explainable AI In Education: Current Trends, Challenges, And Opportunities. *SoutheastCon 2023* (pp. 232-239). Orlando, FL, USA: IEEE. https://doi.org/10.1109/SoutheastCon51012.2023.10115140

Rawal, A., McCoy, J., Rawat, D. B., Sadler, B. M., & Amant, R. S. (2021). Recent Advances in Trustworthy Explainable Artificial Intelligence: Status, Challenges, and Perspectives. *IEEE Transactions on Artificial Intelligence, 3*(6), 852-866. https://doi.org/10.1109/TAI.2021.3133846

Rawat, S., Tyagi, U., & Singhal, S. (2021). Recommender Systems in E-commerce and their Challenges. In *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)* (pp. 1598-

1601). Greater Noida, India: IEEE. https://doi.org/10.1109/ICAC3N53548.2021.9725681

Reddy, S., Nalluri, S., Kunisetti, S., Ashok, S., & Venkatesh, B. (2019). Content-Based Movie Recommendation System Using Genre Correlation. In *Smart Intelligent Computing and Applications. Smart Innovation, Systems and Technologies* (Vol. 105, pp. 391-397). Singapore: Springer. https://doi.org/10.1007/978-981-13-1927-3_42

Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the ACM, 40*(3), 56-58. https://dl.acm.org/doi/pdf/10.1145/245108.245121

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. *In Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW '94)* (pp. 175-186). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/192844.192905

Rosenfeld, A. (2021). Better Metrics for Evaluating Explainable Artificial Intelligence. *In Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '21)* (pp. 45-50). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://dl.acm.org/doi/abs/10.5555/3463952.3463962

Rostami, M., Oussalah, M., & Farrahi, V. (2022). A Novel Time-Aware Food Recommender-System Based on Deep Learning and Graph Clustering. *IEEE Access, 10*, 52508-52524. https://doi.org/10.1109/ACCESS.2022.3175317

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence, 1*(5), 206-215. https://doi.org/10.1038/s42256-019-0048-x

Russell, S. J. (2010). *Artificial Intelligence A Modern Approach* (1 ed.). United Kingdom: Pearson Education, Inc. https://scholar.alaqsa.edu.ps/9195/1/Artificial%20Intelligence%20A%20Modern%20Approach%20%283rd%20Edition%29.pdf%20%28%20PDFDrive%20%29.pdf

Rutjes, H., Willemsen, M., & IJsselsteijn, W. (2019). Considerations on explainable AI and users' mental models. In *CHI 2019 Workshop: Where is the Human? Bridging the Gap Between AI and HCI* (p. 5). Glasgow, Scotland UK: Association for Computing Machinery, Inc. http://www.martijnwillemsen.nl/recommenderlab/RutjesChHI2019ws.pdf

Sahu, M., & Saritha, K. (2021). Study on Various Collabrative Filtering Techniques to Recommend Movies. *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)* (pp. 414-420). Bhopal, India: IEEE. https://doi.org/10.1109/CSNT51715.2021.9509623

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *In Proceedings of the 10th international conference on World Wide Web (WWW '01)* (pp. 285-295). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/371920.372071

Schedl, M. (2016). The LFM-1b Dataset for Music Retrieval and Recommendation. *In Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR '16)* (pp. 103-110). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2911996.2912004

Schröder, G., Thiele, M., & Lehner, W. (2011). Setting goals and choosing metrics for recommender system evaluations. *In UCERSTI2 workshop at the 5th ACM conference on recommender systems. 23*, p. 53. Chicago, USA: ACM. https://ceur-ws.org/Vol-811/paper12.pdf

Sha, X., Sun, Z., & Zhang, J. (2021). Hierarchical attentive knowledge graph embedding for personalized recommendation. *Electronic Commerce Research and Applications*, *48*, 101071. https://doi.org/https://doi.org/10.1016/j.elerap.2021.101071

Sharma, S., Sharma, A., Sharma, Y., & Bhatia, M. (2016). Recommender system using hybrid approach. *2016 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 219-223). Greater Noida, India: IEEE. https://doi.org/10.1109/CCAA.2016.7813722

Shi, C., Hu, B., Zhao, W. X., & Philip, S. Y. (2019). Heterogeneous Information Network Embedding for Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *31*(2), 357-370. https://doi.org/10.1109/TKDE.2018.2833443

Shi, G., Yang, M., & Gao, D. (2022). A novel intrinsically explainable model with semantic manifolds established via transformed priors. *Knowledge-Based Systems*, *252*, 109386. https://doi.org/https://doi.org/10.1016/j.knosys.2022.109386

Shi, Y., Larson, M., & Hanjalic, A. (2014). Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges. *ACM Computing Surveys (CSUR), 47*(1), 1-45. https://doi.org/10.1145/2556270

Silveira, T., Zhang, M., Lin, X., Liu, Y., & Ma, S. (2019). How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics, 10*(5), 813-831. https://doi.org/10.1007/s13042-017-0762-9

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. V., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., & Lanctot, M. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature, 529*(7587), 484-489. https://doi.org/10.1038/nature16961

Singhal, A. (2012). *Introducing the Knowledge Graph: things, not strings.* https://blog.google/products/search/introducing-knowledge-graph-things-not/

Song, W., Duan, Z., Yang, Z., Zhu, H., Zhang, M., & Tang, J. (2019). Explainable knowledge graph-based recommendation via deep reinforcement learning. *arXiv preprint arXiv:1906.09506, 13*. https://www.researchgate.net/profile/Jian-Tang-46/publication/333993762_Explainable_Knowledge_Graph-based_Recommendation_via_Deep_Reinforcement_Learning/links/60728969a6fdcc5f779843f5/Explainable-Knowledge-Graph-based-Recommendation-via-Deep-Reinforcement

Song, W., Wang, T., & Zhang, Z. (2023). Recommendations Based on Reinforcement Learning and Knowledge Graph. In *Advances and Trends in Artificial Intelligence. Theory and Applications: 36th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2023, Shanghai, China, July 19–22, 2023, Proceedings, Part I* (pp. 313–324). Shanghai, China: Springer-Verlag. https://doi.org/10.1007/978-3-031-36819-6_28

Song, Z., Zhang, D., Shi, X. a., Ma, C., & Wu, L. (2021). DEN-DQL: Quick Convergent Deep Q-Learning with Double Exploration Networks for News Recommendation. *2021 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). Shenzhen, China: IEEE. https://doi.org/10.1109/IJCNN52387.2021.9533818

Srivastava, G., Jhaveri, R. H., Bhattacharya, S., Pandya, S., Maddikunta, P. K., Yenduri, G., Hall, J. G., Alazab, M., Gadekallu, T. R., & others. (2022). XAI for cybersecurity: state of the art, challenges, open issues and future directions. *arXiv preprint arXiv:2206.03585*. https://arxiv.org/abs/2206.03585

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence, 2009*, 421425. https://doi.org/10.1155/2009/421425

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). Yago: a core of semantic knowledge. *In Proceedings of the 16th international conference on World Wide Web (WWW '07)* (pp. 697-706). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1242572.1242667

Sun, Y., Han, J., Yan, X., Yu, P. S., & Wu, T. (2011). PathSim: meta path-based top-K similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment, 4*(11), 992-1003. https://doi.org/10.14778/3402707.3402736

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019). RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *arXiv preprint arXiv:1902.10197*. https://arxiv.org/abs/1902.10197

Sun, Z., Guo, Q., Yang, J., Fang, H., Guo, G., Zhang, J., & Burke, R. (2019). Research commentary on recommendations with side information: A survey and research directions. *Electronic Commerce Research and Applications, 37*, 100879. https://doi.org/https://doi.org/10.1016/j.elerap.2019.100879

Sun, Z., Yang, J., Zhang, J., Bozzon, A., Huang, L.-K., & Xu, C. (2018). Recurrent knowledge graph embedding for effective recommendation. *In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)* (pp. 297-305). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3240323.3240361

Sun, Z., Zhang, Q., Hu, W., Wang, C., Chen, M., Akrami, F., & Li, C. (2020). A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs. *arXiv preprint arXiv:2003.07743*. https://arxiv.org/abs/2003.07743

Tang, X., Wang, T., Yang, H., & Song, H. (2019). AKUPM: Attention-Enhanced Knowledge-Aware User Preference Model for Recommendation. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)* (pp. 1891-1899). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3292500.3330705

Tessier, D. (2020). The Needle in the Haystack: How Information Overload Is Impacting Society and Our Search for Truth. In K. Dalkir, & R. Katz (Eds.), *Navigating Fake News, Alternative Facts, and Misinformation in a Post-Truth World* (pp. 18-35). Hershey, PA: IGI Global. https://doi.org/10.4018/978-1-7998-2543-2.ch002

Tintarev, N., & Masthoff, J. (2007). A Survey of Explanations in Recommender Systems. *2007 IEEE 23rd International Conference on Data Engineering Workshop* (pp. 801-810). Istanbul, Turkey: IEEE. https://doi.org/10.1109/ICDEW.2007.4401070

Troussas, C., & Krouska, A. (2022). Path-Based Recommender System for Learning Activities Using Knowledge Graphs. *Information, 14*(1), 9. https://doi.org/10.3390/info14010009

Tsurel, D., Pelleg, D., Guy, I., & Shahaf, D. (2017). Fun Facts: Automatic Trivia Fact Extraction from Wikipedia. *In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)* (pp. 345-354). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3018661.3018709

Vale, D., El-Sharif, A., & Ali, M. (2022). Explainable artificial intelligence (XAI) post-hoc explainability methods: Risks and limitations in non-discrimination law. *AI Ethics, 2*(4), 815-826. https://doi.org/10.1007/s43681-022-00142-y

Voigt, P., & Von dem Bussche, A. (2017). *The EU General Data Protection Regulation (GDPR)* (1 ed., Vol. 10). Springer. https://doi.org/10.1007/978-3-319-57959-7

Vrandečić, D., & Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM, 57,* 78--85. https://doi.org/10.1145/2629489

Vultureanu-Albişi, A., & Bădică, C. (2021). Recommender Systems: An Explainable AI Perspective. *2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)* (pp. 1-6). Kocaeli, Turkey: IEEE. https://doi.org/10.1109/INISTA52262.2021.9548125

Wan, M. a. (2018). Item recommendation on monotonic behavior chains. *In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)* (pp. 86-94). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3240323.3240369

Wang, C., Zhu, H., Zhu, C., Qin, C., & Xiong, H. (2020). SetRank: A Setwise Bayesian Approach for Collaborative Ranking from Implicit Feedback. *Proceedings of the AAAI Conference on Artificial Intelligence. 34*, pp. 6127-6136. New York, USA: AAAI Press, Palo Alto, California USA. https://doi.org/10.1609/aaai.v34i04.6077

Wang, D., Liang, Y., Xu, D., Feng, X., & Guan, R. (2018). A content-based recommender system for computer science publications. *Knowledge-Based Systems, 157*, 1-9. https://doi.org/https://doi.org/10.1016/j.knosys.2018.05.001

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2018). RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. *In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)* (pp. 417-426). Torino, Italy: Association for Computing Machinery. https://doi.org/10.1145/3269206.3271739

Wang, H., Zhang, F., Xie, X., & Guo, M. (2018a). DKN: Deep Knowledge-Aware Network for News Recommendation. *In Proceedings of the 2018 World Wide Web Conference (WWW '18)* (pp. 1835-1844). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/3178876.3186175

Wang, H., Zhao, M., Xie, X., Li, W., & Guo, M. (2019). Knowledge Graph Convolutional Networks for Recommender Systems. *In The World Wide Web Conference (WWW '19)* (pp. 3307--3313). San Francisco, CA, USA: Association for Computing Machinery. https://doi.org/10.1145/3308558.3313417

Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering, 29*(12), 2724-2743. https://doi.org/10.1109/TKDE.2017.2754499

Wang, X., Chen, Y., Yang, J., Wu, L., Wu, Z., & Xie, X. (2018). A Reinforcement Learning Framework for Explainable Recommendation. *2018 IEEE*

International Conference on Data Mining (ICDM) (pp. 587-596). Singapore: IEEE. https://doi.org/10.1109/ICDM.2018.00074

Wang, X., He, X., Cao, Y., Liu, M., & Chua, T.-S. (2019). KGAT: Knowledge Graph Attention Network for Recommendation. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)* (pp. 950--958). Anchorage, AK, USA: Association for Computing Machinery. https://doi.org/10.1145/3292500.3330989

Wang, X., Liu, K., Wang, D., Wu, L., Fu, Y., & Xie, X. (2022). Multi-level Recommendation Reasoning over Knowledge Graphs with Reinforcement Learning. *In Proceedings of the ACM Web Conference 2022 (WWW '22)* (pp. 2098-2108). Virtual Event, Lyon, France: Association for Computing Machinery. https://doi.org/10.1145/3485447.3512083

Wang, X., Wang, D., Xu, C., He, X., Cao, Y., & Chua, T.-S. (2019). Explainable Reasoning over Knowledge Graphs for Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence. 33*, pp. 5329-5336. Honolulu, Hawaii, USA: AAAI Press. https://doi.org/10.1609/aaai.v33i01.33015329

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). Knowledge Graph Embedding by Translating on Hyperplanes. *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence,. 28*, pp. 1112-1119. Québec City, Québec, Canada: AAAI Press. https://doi.org/10.1609/aaai.v28i1.8870

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning, 8*(3), 279-292. https://doi.org/10.1007/BF00992698

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning, 8*(3), 229-256. https://doi.org/10.1007/BF00992696

Witten, E. (2020). A mini-introduction to information theory. *La Rivista del Nuovo Cimento, 43*(4), 187-227. https://doi.org/10.1007/s40766-020-00004-5

Wu, F., Qiao, Y., Chen, J.-H., Wu, C., Qi, T., Lian, J., Liu, D., Xie, X., Gao, J., Wu, W., & others. (2020). Mind: A Large-scale Dataset for News Recommendation. *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 3597-3606). Online: Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.331

Wu, S., Sun, F., Zhang, W., Xie, X., & Cui, B. (2022, May). Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys, 55*(5), 1-37. https://doi.org/10.1145/3535101

Wu, Y., Macdonald, C., & Ounis, I. (2020). A Hybrid Conditional Variational Autoencoder Model for Personalised Top-n Recommendation. *In Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval (ICTIR '20)* (pp. 89-96). Virtual Event, Norway: Association for Computing Machinery. https://doi.org/10.1145/3409256.3409835

Xi, D., Zhuang, F., Liu, Y., Gu, J., Xiong, H., & He, Q. (2019). Modelling of Bi-Directional Spatio-Temporal Dependence and Users' Dynamic Preferences for Missing POI Check-In Identification. *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence. 33*, pp. 5458-5465. Honolulu, Hawaii, USA: AAAI Press. https://doi.org/10.1609/aaai.v33i01.33015458

Xian, Y., Fu, Z., De Melo, G., & Zhang, Y. (2019). Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. *In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)* (pp. 285-294). Paris, France: Association for Computing Machinery. https://doi.org/10.1145/3331184.3331203

Xian, Y., Fu, Z., Muthukrishnan, S., de Melo, G., & Zhang, Y. (2019). Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. In *SIGIR'19: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval* (pp. 285–294, doi: https://doi.org/10.1145/3331184.3331203). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3331184.3331203

Xie, C. (2023). Commodity knowledge graph-based TransD-KGAT method for recommendation. In Y. Zhong (Ed.), *Fifth International Conference on Computer Information Science and Artificial Intelligence (CISAI 2022). 12566*, p. 125662F. Chongqing, China: SPIE. https://doi.org/10.1117/12.2667700

Xiong, W., Hoang, T., & Wang, W. Y. (2017). DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (pp. 564-573). Copenhagen, Denmark: Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1060

Xu, W., Gao, X., Sheng, Y., & Chen, G. (2021). Recommendation System with Reasoning Path Based on DQN and Knowledge Graph. *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)* (pp. 1-8). Seoul, Korea (South): IEEE. https://doi.org/10.1109/IMCOM51814.2021.9377414

Xu, Y., Wang, E., Yang, Y., & Chang, Y. (2021). A Unified Collaborative Representation Learning for Neural-Network Based Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering, 34*(11), 5126-5139. https://doi.org/10.1109/TKDE.2021.3054782

Xu, Y., Yang, Y., Han, J., Wang, E., Zhuang, F., & Xiong, H. (2018). Exploiting the Sentimental Bias between Ratings and Reviews for Enhancing Recommendation. *2018 IEEE International Conference on Data Mining (ICDM)* (pp. 1356-1361). Singapore: IEEE. https://doi.org/10.1109/ICDM.2018.00185

Yahya, M., Breslin, J. G., & Ali, M. I. (2021). Semantic Web and Knowledge Graphs for Industry 4.0. *Applied Sciences, 11*(11), 5110. https://doi.org/10.3390/app11115110

Yan, Z., Ge, J., Wu, Y., Li, L., & Li, T. (2020, June). Automatic Virtual Network Embedding: A Deep Reinforcement Learning Approach With Graph Convolutional Networks. *IEEE Journal on Selected Areas in Communications, 38*(6), 1040-1057. https://doi.org/10.1109/JSAC.2020.2986662.

Yang, B., Yih, S. W.-t., He, X., Gao, J., & Deng, L. (2015). Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *Proceedings of the International Conference on Learning Representations (ICLR) 2015.* https://www.microsoft.com/en-us/research/publication/embedding-entities-and-relations-for-learning-and-inference-in-knowledge-bases/

Yang, C., Yu, X., Liu, Y., Nie, Y., & Wang, Y. (2016, October 19). Collaborative filtering with weighted opinion aspects. *Neurocomputing, 210*, 185-196. https://doi.org/https://doi.org/10.1016/j.neucom.2015.12.136

Yang, D., Guo, Z., Wang, Z., Jiang, J., Xiao, Y., & Wang, W. (2018). A Knowledge-Enhanced Deep Recommendation Framework Incorporating GAN-Based Models. *2018 IEEE International Conference on Data Mining (ICDM) 17-20 Nov. 2018* (pp. 1368-1373). Singapore: IEEE. https://doi.org/10.1109/ICDM.2018.00187

Yang, X., Du, X., & Wang, M. (2020). Learning to Match on Graph for Fashion Compatibility Modeling. *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence. 34*, pp. 287-294. New York, USA: AAAI Press. https://doi.org/10.1609/aaai.v34i01.5362

Ye, Y., Wang, X., Yao, J., Jia, K., Zhou, J., Xiao, Y., & Yang, H. (2019). Bayes EMbedding (BEM): Refining Representation by Integrating Knowledge Graphs and Behavior-specific Networks. *In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)* (pp. 679-688). Beijing, China: Association for Computing Machinery. https://doi.org/10.1145/3357384.3358014

Yelp. (2013). *Yelp challenge dataset*. https://www.kaggle.com/c/yelp-recsys-2013/

Yin, M. W. (2019). Understanding the Effect of Accuracy on Trust in Machine Learning Models. *In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)* (pp. 1-12). Glasgow, Scotland Uk: Association for Computing Machinery. https://doi.org/10.1145/3290605.3300509

Yin, Y., Chen, L., Wan, J., & others. (2018). Location-Aware Service Recommendation With Enhanced Probabilistic Matrix Factorization. *IEEE Access, 6*, 62815-62825. https://doi.org/10.1109/ACCESS.2018.2877137

Yu, D., Zhu, C., Yang, Y., & Zeng, M. (2022). JAKET: Joint Pre-training of Knowledge Graph and Language Understanding. *Proceedings of the Thirty-*

Sixth AAAI Conference on Artificial Intelligence. 36, pp. 11630-11638. Virtual: AAAI Press. https://doi.org/10.1609/aaai.v36i10.21417

Yu, X., Ren, X., Gu, Q., Sun, Y., & Han, J. (2013). Collaborative filtering with entity similarity regularization in heterogeneous information networks. *IJCAI HINA, 27*. http://hanj.cs.illinois.edu/pdf/hina13_xyu.pdf

Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., & Han, J. (2014). Personalized entity recommendation: A heterogeneous information network approach. *In Proceedings of the 7th ACM international conference on Web search and data mining (WSDM '14)* (pp. 283-292). New York, USA: Association for Computing Machinery. https://doi.org/10.1145/2556195.2556259

Zhang, F., Yuan, N. J., Lian, D., Xie, X., & Ma, W.-Y. (2016). Collaborative Knowledge Base Embedding for Recommender Systems. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)* (pp. 353-362). San Francisco, California, USA: Association for Computing Machinery. https://doi.org/10.1145/2939672.2939673

Zhang, L. (2013, June). The Definition of Novelty in Recommendation System. *Journal of Engineering Science & Technology Review, 6*(3), 141-145. http://jestr.org/downloads/Volume6Issue3/fulltext25632013.pdf

Zhang, S., Ouyang, Y., Liu, Z., Rong, W., & Xiong, Z. (2022). Reinforcement Learning-Based Explainable Recommendation over Knowledge Graphs with Negative Sampling. In *2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)* (pp. 1948-1953, doi: 10.1109/SmartWorld-UIC-ATC-ScalCom-DigitalTwin-PriComp-Metaverse56740.2022.00282). Haikou, China: IEEE. https://doi.org/10.1109/SmartWorld-UIC-ATC-ScalCom-DigitalTwin-PriComp-Metaverse56740.2022.00282

Zhang, W., Yuan, Q., Han, J., & Wang, J. (2016). Collaborative multi-level embedding learning from reviews for rating prediction. *In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16). 16*, pp. 2986-2992. New York, USA: AAAI Press. https://doi.org/abs/10.5555/3060832.3061039

Zhang, Y., & Chen, X. (2020, March 11). Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval, 14*(1), 1-101. https://doi.org/http://dx.doi.org/10.1561/1500000066

Zhang, Y., Ai, Q., Chen, X., & Croft, W. B. (2017). Joint Representation Learning for Top-N Recommendation with Heterogeneous Information Sources. *Association for Computing Machinery* (pp. 1449--1458). Singapore:

Association for Computing Machinery. https://doi.org/10.1145/3132847.3132892

Zhang, Y., Ai, Q., Chen, X., & Wang, P. (2018). Learning over knowledge-base embeddings for recommendation. *Algorithms 11(9):137, 2018, Special Issue on Collaborative Filtering and Recommender Systems, 11*(9), 137. https://doi.org/10.48550/arXiv.1803.06540

Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., & Ma, S. (2014a). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. *In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (SIGIR '14)* (pp. 83-92). Gold Coast, Queensland, Australia: Association for Computing Machinery. https://doi.org/10.1145/2600428.2609579

Zhang, Y., Teoh, B. K., Wu, M., Chen, J., & Zhang, L. (2023, January 1). Data-driven estimation of building energy consumption and GHG emissions using explainable artificial intelligence. *Energy, 262*, 125468. https://doi.org/https://doi.org/10.1016/j.energy.2022.125468

Zhang, Z., Zhang, Y., & Ren, Y. (2020, August). Employing neighborhood reduction for alleviating sparsity and cold start problems in user-based collaborative filtering}. *Information Retrieval Journal, 23*, 449-472. https://doi.org/10.1007/s10791-020-09378-w

Zhao, H., Yao, Q., Li, J., Song, Y., & Lee, D. L. (2017). Meta-Graph Based Recommendation Fusion over Heterogeneous Information Networks. *In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)* (pp. 635-644). Halifax, NS, Canada: Association for Computing Machinery. https://doi.org/10.1145/3097983.3098063

Zhao, J., Zhou, Z., Guan, Z., Zhao, W., Ning, W., Qiu, G., & He, X. (2019). IntentGC: A Scalable Graph Convolution Framework Fusing Heterogeneous Information for Recommendation. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)* (pp. 2347-2357). Anchorage, AK, USA: Association for Computing Machinery. https://doi.org/10.1145/3292500.3330686

Zhao, K., Wang, X., Zhang, Y., Zhao, L., Liu, Z., Xing, C., & Xie, X. (2020). Leveraging Demonstrations for Reinforcement Recommendation Reasoning over Knowledge Graphs. In *SIGIR '20: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 239–248, doi: https://doi.org/10.1145/3397271.3401171). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3397271.3401171

Zheng, L., Noroozi, V., & Yu, P. S. (2017). Joint Deep Modeling of Users and Items Using Reviews for Recommendation. *In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)* (pp.

425-434). Cambridge, United Kingdom: Association for Computing Machinery. https://doi.org/10.1145/3018661.3018665

Zheng, W., Yin, L., Chen, X., Ma, Z., Liu, S., & Yang, B. (2021). Knowledge base graph embedding module design for Visual question answering model. *Pattern Recognition, 120*, 108153. https://doi.org/https://doi.org/10.1016/j.patcog.2021.108153

Zhuang, F., Zhang, Z., Qian, M., Shi, C., Xie, X., & He, Q. (2017). Representation learning via dual-autoencoder for recommendation. *Neural Networks, 90*, 83-89. https://doi.org/https://doi.org/10.1016/j.neunet.2017.03.009

Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. *In Proceedings of the 14th international conference on World Wide Web (WWW '05)* (pp. 22-32). Chiba, Japan: Association for Computing Machinery. https://doi.org/10.1145/1060745.1060754

Zou, X. (2020, March). A Survey on Application of Knowledge Graph. *Journal of Physics: Conference Series, 1487*(1), 012016. https://doi.org/10.1088/1742-6596/1487/1/012016

APPENDIX A

EXPERIMENT RESULTS - BEAUTY

(1) KGE Model – Execution statistics

(First iteration of every epoch)

[INFO] Namespace(dataset='beauty', name='train_transe_model', seed=123, gpu='1', epochs=30, batch_size=64, lr=0.5, weight_decay=0, l2_lambda=0, max_grad_norm=5.0, embed_size=100, num_neg_samples=5, steps_per_checkpoint=10000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device='cpu', dir='./tmp/Amazon_Beauty/train_transe_model', checkpoint_dir='./tmp/Amazon_Beauty/train_transe_model/checkpoint', log_dir='./tmp/Amazon_Beauty/train_transe_model/log')

[INFO] Parameters:['purchase', 'mentions', 'describe_as', 'produced_by', 'belongs_to', 'also_bought', 'also_viewed', 'bought_together', 'user.weight', 'product.weight', 'word.weight', 'related_product.weight', 'brand.weight', 'category.weight', 'purchase_bias.weight', 'mentions_bias.weight', 'describe_as_bias.weight', 'produced_by_bias.weight', 'belongs_to_bias.weight', 'also_bought_bias.weight', 'also_viewed_bias.weight', 'bought_together_bias.weight']

[INFO] Epoch: 01 | Words: 1655772/405897991 | Lr: 0.49796 | Smooth loss: 21.18494

[INFO] Epoch: 02 | Words: 14880496/405897991 | Lr: 0.48167 | Smooth loss: 12.49645

[INFO] Epoch: 03 | Words: 28098180/405897991 | Lr: 0.46539 | Smooth loss: 11.30226

[INFO] Epoch: 04 | Words: 41320119/405897991 | Lr: 0.44910 | Smooth loss: 10.59622

[INFO] Epoch: 05 | Words: 54544681/405897991 | Lr: 0.43281 | Smooth loss: 10.15674

[INFO] Epoch: 06 | Words: 67768114/405897991 | Lr: 0.41652 | Smooth loss: 9.79851

[INFO] Epoch: 07 | Words: 82652088/405897991 | Lr: 0.39819 | Smooth loss: 9.17978

[INFO] Epoch: 08 | Words: 95875721/405897991 | Lr: 0.38190 | Smooth loss: 8.98994

[INFO] Epoch: 09 | Words: 109102468/405897991 | Lr: 0.36560 | Smooth loss: 8.75285

[INFO] Epoch: 10 | Words: 122327060/405897991 | Lr: 0.34931 | Smooth loss: 8.60673

[INFO] Epoch: 11 | Words: 135551066/405897991 | Lr: 0.33302 | Smooth loss: 8.45182

[INFO] Epoch: 12 | Words: 150427158/405897991 | Lr: 0.31470 | Smooth loss: 8.14496

[INFO] Epoch: 13 | Words: 163643939/405897991 | Lr: 0.29842 | Smooth loss: 7.94083

[INFO] Epoch: 14 | Words: 176867579/405897991 | Lr: 0.28213 | Smooth loss: 7.84264

[INFO] Epoch: 15 | Words: 190769625/405897991 | Lr: 0.26500 | Smooth loss: 7.52107

[INFO] Epoch: 16 | Words: 203987309/405897991 | Lr: 0.24872 | Smooth loss: 7.42598

[INFO] Epoch: 17 | Words: 217209248/405897991 | Lr: 0.23243 | Smooth loss: 7.26999

[INFO] Epoch: 18 | Words: 230433810/405897991 | Lr: 0.21614 | Smooth loss: 7.17050

[INFO] Epoch: 19 | Words: 244889357/405897991 | Lr: 0.19834 | Smooth loss: 6.69575

[INFO] Epoch: 20 | Words: 258107041/405897991 | Lr: 0.18205 | Smooth loss: 6.63928

[INFO] Epoch: 21 | Words: 271328980/405897991 | Lr: 0.16577 | Smooth loss: 6.53380

[INFO] Epoch: 22 | Words: 284553542/405897991 | Lr: 0.14948 | Smooth loss: 6.57166

[INFO] Epoch: 23 | Words: 297776975/405897991 | Lr: 0.13319 | Smooth loss: 6.68610

[INFO] Epoch: 24 | Words: 312660949/405897991 | Lr: 0.11485 | Smooth loss: 6.50444

[INFO] Epoch: 25 | Words: 325884582/405897991 | Lr: 0.09856 | Smooth loss: 6.43622

[INFO] Epoch: 26 | Words: 339111329/405897991 | Lr: 0.08227 | Smooth loss: 6.32593

[INFO] Epoch: 27 | Words: 352335921/405897991 | Lr: 0.06598 | Smooth loss: 6.26586

[INFO] Epoch: 28 | Words: 365559927/405897991 | Lr: 0.04969 | Smooth loss: 6.19449

[INFO] Epoch: 29 | Words: 380436019/405897991 | Lr: 0.03137 | Smooth loss: 6.11547

[INFO] Epoch: 30 | Words: 393652800/405897991 | Lr: 0.01508 | Smooth loss: 6.02995

(2) RL Model – Train - Execution Statistics

(First Iteration of Every Epoch – Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='beauty', name='train_RL_agent', seed=123, gpu='0', epochs=100, batch_size=32, lr=0.0001, max_acts=250, max_path_len=3, gamma=0.99, ent_weight=0.001, act_dropout=0, state_history=1, hidden=[512, 256], debug=0, steps_per_checkpoint=5000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=0, logging_mode='a', device=device(type='cuda', index=0), dir='./tmp/Amazon_Beauty/train_RL_agent', checkpoint_dir='./tmp/Amazon_Beauty/train_RL_agent/checkpoint', log_dir='./tmp/Amazon_Beauty/train_RL_agent/log')

[INFO] Parameters:['l1.weight', 'l1.bias', 'l2.weight', 'l2.bias', 'actor.weight', 'actor.bias', 'critic.weight', 'critic.bias']

[INFO] epoch/step=1/5000 | loss=36.70607 | ploss=36.71131 | vloss=520952.85223 | entropy=-5.24462 | reward=11.63741

[INFO] Save model to ./tmp/Amazon_Beauty/train_RL_agent/checkpoint/policy_model_epoch_1.ckpt

[INFO] epoch/step=2/40000 | loss=3.68918 | ploss=3.69375 | vloss=30046316.88555 | entropy=-4.56614 | reward=306.27913

[INFO] Save model to ./tmp/Amazon_Beauty/train_RL_agent/checkpoint/policy_model_epoch_2.ckpt

[INFO] epoch/step=3/125000 | loss=0.93528 | ploss=0.94039 | vloss=16945203.40255 | entropy=-5.10520 | reward=170.46549

[INFO] Save model to ./tmp/Amazon_Beauty/train_RL_agent/checkpoint/policy_model_epoch_3.ckpt

[INFO] epoch/step=4/245000 | loss=0.87873 | ploss=0.88388 | vloss=9970487.33428 | entropy=-5.15191 | reward=102.23704

[INFO] Save model to ./tmp/Amazon_Beauty/train_RL_agent/checkpoint/policy_model_epoch_4.ckpt

[INFO] epoch/step=5/370000 | loss=0.77097 | ploss=0.77612 | vloss=13609808.61651 | entropy=-5.15190 | reward=138.68565

[INFO] Save model to ./tmp/Amazon_Beauty/train_RL_agent/checkpoint/policy_model_epoch_5.ckpt

(3) RL Model – Test - Execution Statistics

(Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='beauty', source_name='train_RL_agent', output_folder='test_RL_agent', seed=123, gpu='0', epochs=100, max_acts=250, max_path_len=3, gamma=0.99, state_history=1, hidden=[512, 256], add_products=False, topk=[10, 10, 12], run_path=True, run_eval=True, debug=0, batch_size=512, is_resume_from_checkpoint=0, logging_mode='a', log_file_name='test_agent_log', checkpoint_folder='checkpoint', explainability_score_option=1, run_number='1', is_only_run_specific_epoch=0, device=device(type='cuda', index=0), output_dir='./tmp/Amazon_Beauty/test_RL_agent')

[INFO] model epoch=1 | count (users)=22363 | ndcg=6.49714 | recall=10.22396 | hit_ratio=17.06095 | precision=2.12602 | invalid_users=2.00000 | execution_timestamp=2023-07-04 20:30:21.537342

[INFO] model epoch=2 | count (users)=22363 | ndcg=6.59601 | recall=10.23038 | hit_ratio=17.01623 | precision=2.12289 | invalid_users=2.00000 | execution_timestamp=2023-07-04 22:00:17.421091

[INFO] model epoch=3 | count (users)=22363 | ndcg=6.60099 | recall=10.32048 | hit_ratio=17.19064 | precision=2.14481 | invalid_users=2.00000 | execution_timestamp=2023-07-04 23:29:21.637353

[INFO] model epoch=4 | count (users)=22363 | ndcg=6.60477 | recall=10.30196 | hit_ratio=17.21300 | precision=2.14436 | invalid_users=2.00000 | execution_timestamp=2023-07-05 00:58:19.011753

[INFO] model epoch=5 | count (users)=22363 | ndcg=6.53770 | recall=10.20221 | hit_ratio=17.00729 | precision=2.11529 | invalid_users=2.00000 | execution_timestamp=2023-07-05 02:26:58.074233

APPENDIX B

EXPERIMENT RESULTS – CD & VINYL

(1) KGE Model – Execution Statistics

(First Iteration of Every Epoch)

[INFO] Namespace(dataset='cd', name='train_transe_model', seed=123, gpu='1', epochs=30, batch_size=256, lr=0.5, weight_decay=0, l2_lambda=0, max_grad_norm =5.0, embed_size=100, num_neg_samples=5, steps_per_checkpoint=10000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device='cpu', dir='./tmp /Amazon_CDs/train_transe_model', checkpoint_dir='./tmp/Amazon_CDs/ train_transe_model/checkpoint', log_dir='./tmp/Amazon_CDs/train_transe_model/log')

[INFO] Parameters:['purchase', 'mentions', 'describe_as', 'produced_by', 'belongs_to', 'also_bought', 'also_viewed', 'bought_together', 'user.weight', 'product.weight', 'word.weight', 'related_product.weight', 'brand.weight', 'category.weight', 'purchase_bias.weight', 'mentions_bias.weight', 'describe_as_bias.weight', 'produced_by_bias.weight', 'belongs_to_bias.weight', 'also_bought_bias.weight', 'also_viewed_bias.weight', 'bought_together_bias.weight']

[INFO] Epoch: 01 | Words: 5513023/4169777431 | Lr: 0.49934 | Smooth loss: 20.77711

[INFO] Epoch: 02 | Words: 143369396/4169777431 | Lr: 0.48281 | Smooth loss: 13.10005

[INFO] Epoch: 03 | Words: 281224411/4169777431 | Lr: 0.46628 | Smooth loss: 11.76092

[INFO] Epoch: 04 | Words: 421353988/4169777431 | Lr: 0.44948 | Smooth loss: 10.95059

[INFO] Epoch: 05 | Words: 564725794/4169777431 | Lr: 0.43228 | Smooth loss: 10.43205

[INFO] Epoch: 06 | Words: 697056168/4169777431 | Lr: 0.41642 | Smooth loss: 10.16354

[INFO] Epoch: 07 | Words: 840425487/4169777431 | Lr: 0.39922 | Smooth loss: 9.80237

[INFO] Epoch: 08 | Words: 977324312/4169777431 | Lr: 0.38281 | Smooth loss: 9.10926

[INFO] Epoch: 09 | Words: 1120696118/4169777431 | Lr: 0.36562 | Smooth loss: 8.89732

[INFO] Epoch: 10 | Words: 1253026492/4169777431 | Lr: 0.34975 | Smooth loss: 8.75423

[INFO] Epoch: 11 | Words: 1396395811/4169777431 | Lr: 0.33256 | Smooth loss: 8.91138

[INFO] Epoch: 12 | Words: 1539772574/4169777431 | Lr: 0.31537 | Smooth loss: 8.90235

[INFO] Epoch: 13 | Words: 1672117200/4169777431 | Lr: 0.29950 | Smooth loss: 8.75983

[INFO] Epoch: 14 | Words: 1815464969/4169777431 | Lr: 0.28231 | Smooth loss: 8.56184

[INFO] Epoch: 15 | Words: 1947827741/4169777431 | Lr: 0.26644 | Smooth loss: 8.45959

[INFO] Epoch: 16 | Words: 2089264960/4169777431 | Lr: 0.24948 | Smooth loss: 7.88948

[INFO] Epoch: 17 | Words: 2228257541/4169777431 | Lr: 0.23281 | Smooth loss: 7.62325

[INFO] Epoch: 18 | Words: 2371629347/4169777431 | Lr: 0.21562 | Smooth loss: 7.71872

[INFO] Epoch: 19 | Words: 2503959721/4169777431 | Lr: 0.19975 | Smooth loss: 7.60287

[INFO] Epoch: 20 | Words: 2647329040/4169777431 | Lr: 0.18256 | Smooth loss: 7.61598

[INFO] Epoch: 21 | Words: 2790705803/4169777431 | Lr: 0.16537 | Smooth loss: 7.60095

[INFO] Epoch: 22 | Words: 2923050429/4169777431 | Lr: 0.14950 | Smooth loss: 7.53192

[INFO] Epoch: 23 | Words: 3060596203/4169777431 | Lr: 0.13300 | Smooth loss: 7.47241

[INFO] Epoch: 24 | Words: 3198443876/4169777431 | Lr: 0.11647 | Smooth loss: 7.44554

[INFO] Epoch: 25 | Words: 3336294084/4169777431 | Lr: 0.09994 | Smooth loss: 7.35387

[INFO] Epoch: 26 | Words: 3479191340/4169777431 | Lr: 0.08281 | Smooth loss: 7.24177

[INFO] Epoch: 27 | Words: 3617046355/4169777431 | Lr: 0.06628 | Smooth loss: 7.20300

[INFO] Epoch: 28 | Words: 3754893984/4169777431 | Lr: 0.04975 | Smooth loss: 7.15392

[INFO] Epoch: 29 | Words: 3892751671/4169777431 | Lr: 0.03322 | Smooth loss: 7.10185

[INFO] Epoch: 30 | Words: 4036129067/4169777431 | Lr: 0.01603 | Smooth loss: 7.04347

(2) RL Model – Train - Execution Statistics

(First Iteration of Every Epoch – Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cd', name='train_RL_agent', seed=123, gpu='0', epochs=100, batch_size=32, lr=0.0001, max_acts=250, max_path_len=3, gamma=0.99, ent_weight=0.001, act_dropout=0, state_history=1, hidden=[512, 256], debug=0, steps_per_checkpoint=50000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device=device(type='cuda', index=0), dir='./tmp/Amazon_CDs/train_RL_agent', checkpoint_dir='./tmp/Amazon_CDs/train_RL_agent/checkpoint', log_dir='./tmp/Amazon_CDs/train_RL_agent/log')

[INFO] Parameters:['l1.weight', 'l1.bias', 'l2.weight', 'l2.bias', 'actor.weight', 'actor.bias', 'critic.weight', 'critic.bias']

[INFO] epoch/step=1/50000 | loss=17.06913 | ploss=17.07434 | vloss=19051380.40819 | entropy=-5.21253 | reward=192.99083

[INFO] Save model to ./tmp/Amazon_CDs/train_RL_agent/checkpoint/policy_model_epoch_1.ckpt

[INFO] epoch/step=2/500000 | loss=-0.33172 | ploss=-0.32641 | vloss=13334084.06803 | entropy=-5.31685 | reward=133.73846

[INFO] Save model to ./tmp/Amazon_CDs/train_RL_agent/checkpoint/policy_model_epoch_2.ckpt

[INFO] epoch/step=3/1950000 | loss=-0.33401 | ploss=-0.32868 | vloss=17863413.85720 | entropy=-5.33209 | reward=178.20004

[INFO] Save model to ./tmp/Amazon_CDs/train_RL_agent/checkpoint/policy_model_epoch_3.ckpt

[INFO] epoch/step=4/6150000 | loss=-0.32634 | ploss=-0.32088 | vloss=3577994.04891 | entropy=-5.46220 | reward=35.83818

[INFO] Save model to ./tmp/Amazon_CDs/train_RL_agent/checkpoint/policy_model_epoch_4.ckpt

[INFO] epoch/step=5/12300000 | loss=-0.30796 | ploss=-0.30248 | vloss=3178170.92674 | entropy=-5.47837 | reward=31.73411

[INFO] Save model to ./tmp/Amazon_CDs/train_RL_agent/checkpoint/policy_model_epoch_5.ckpt

(3) RL Model – Test - Execution Statistics

(Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cd', source_name='train_RL_agent', output_folder='test_RL_agent', seed=123, gpu='0', epochs=100, max_acts=250, max_path_len=3, gamma=0.99, state_history=1, hidden=[512, 256], add_products=False, topk=[10, 10, 12], run_path=True, run_eval=True, debug=0, batch_size=512, is_resume_from_checkpoint=0, logging_mode='a', log_file_name='test_agent_log', checkpoint_folder='checkpoint', explainability_score_option=1, run_number='1', is_only_run_specific_epoch=0, device=device(type='cuda', index=0), output_dir='./tmp/Amazon_CDs/test_RL_agent')

[INFO] model epoch=1 | count (users)=75258 | ndcg=5.56285 | recall=7.96182 | hit_ratio=17.73640 | precision=2.32212 | invalid_users=68.00000 | execution_timestamp=2023-07-01 21:07:50.412794

[INFO] model epoch=2 | count (users)=75258 | ndcg=5.59125 | recall=8.03446 | hit_ratio=17.85477 | precision=2.33781 | invalid_users=68.00000 | execution_timestamp=2023-07-02 05:04:14.941693

[INFO] model epoch=3 | count (users)=75258 | ndcg=5.57115 | recall=8.00161 | hit_ratio=17.78295 | precision=2.33036 | invalid_users=68.00000 | execution_timestamp=2023-07-02 12:41:22.019558

[INFO] model epoch=4 | count (users)=75258 | ndcg=5.56908 | recall=7.99145 | hit_ratio=17.78960 | precision=2.32943 | invalid_users=68.00000 | execution_timestamp=2023-07-02 19:25:41.344565

[INFO] model epoch=5 | count (users)=75258 | ndcg=5.57017 | recall=7.99819 | hit_ratio=17.76832 | precision=2.32917 | invalid_users=68.00000 | execution_timestamp=2023-07-03 02:13:45.092778

APPENDIX C

EXPERIMENT RESULTS – CELL PHONES

(1) KGE Model – Execution Statistics

(First Iteration of Every Epoch)

[INFO] Namespace(dataset='cell', name='train_transe_model', seed=123, gpu='1', epochs=30, batch_size=64, lr=0.5, weight_decay=0, l2_lambda=0, max_grad_norm=5.0, embed_size=100, num_neg_samples=5, steps_per_checkpoint=10000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device='cpu', dir='./tmp/Amazon_Cellphones/train_transe_model', checkpoint_dir='./tmp/Amazon_Cellphones/train_transe_model/checkpoint', log_dir='./tmp/Amazon_Cellphones/train_transe_model/log')

[INFO] Parameters:['purchase', 'mentions', 'describe_as', 'produced_by', 'belongs_to', 'also_bought', 'also_viewed', 'bought_together', 'user.weight', 'product.weight', 'word.weight', 'related_product.weight', 'brand.weight', 'category.weight', 'purchase_bias.weight', 'mentions_bias.weight', 'describe_as_bias.weight', 'produced_by_bias.weight', 'belongs_to_bias.weight', 'also_bought_bias.weight', 'also_viewed_bias.weight', 'bought_together_bias.weight']

[INFO] Epoch: 01 | Words: 1644921/418195831 | Lr: 0.49803 | Smooth loss: 17.94315

[INFO] Epoch: 02 | Words: 14771651/418195831 | Lr: 0.48234 | Smooth loss: 12.50883

[INFO] Epoch: 03 | Words: 27902346/418195831 | Lr: 0.46664 | Smooth loss: 11.85953

[INFO] Epoch: 04 | Words: 42676853/418195831 | Lr: 0.44898 | Smooth loss: 10.89948

[INFO] Epoch: 05 | Words: 55803465/418195831 | Lr: 0.43328 | Smooth loss: 10.54798

[INFO] Epoch: 06 | Words: 70574458/418195831 | Lr: 0.41562 | Smooth loss: 9.98812

[INFO] Epoch: 07 | Words: 83706870/418195831 | Lr: 0.39992 | Smooth loss: 9.72208

[INFO] Epoch: 08 | Words: 98476728/418195831 | Lr: 0.38226 | Smooth loss: 9.42398

[INFO] Epoch: 09 | Words: 112350678/418195831 | Lr: 0.36567 | Smooth loss: 8.96125

[INFO] Epoch: 10 | Words: 125481373/418195831 | Lr: 0.34997 | Smooth loss: 8.87174

[INFO] Epoch: 11 | Words: 140255880/418195831 | Lr: 0.33231 | Smooth loss: 8.61516

[INFO] Epoch: 12 | Words: 155021475/418195831 | Lr: 0.31465 | Smooth loss: 8.32090

[INFO] Epoch: 13 | Words: 168153485/418195831 | Lr: 0.29895 | Smooth loss: 8.23190

[INFO] Epoch: 14 | Words: 181285897/418195831 | Lr: 0.28325 | Smooth loss: 8.07549

[INFO] Epoch: 15 | Words: 196055755/418195831 | Lr: 0.26559 | Smooth loss: 8.10981

[INFO] Epoch: 16 | Words: 209185882/418195831 | Lr: 0.24990 | Smooth loss: 8.11780

[INFO] Epoch: 17 | Words: 223958630/418195831 | Lr: 0.23223 | Smooth loss: 7.91764

[INFO] Epoch: 18 | Words: 237092997/418195831 | Lr: 0.21653 | Smooth loss: 7.83632

[INFO] Epoch: 19 | Words: 253503405/418195831 | Lr: 0.19691 | Smooth loss: 7.65433

[INFO] Epoch: 20 | Words: 264995922/418195831 | Lr: 0.18317 | Smooth loss: 7.57748

[INFO] Epoch: 21 | Words: 281413236/418195831 | Lr: 0.16354 | Smooth loss: 7.41523

[INFO] Epoch: 22 | Words: 292899717/418195831 | Lr: 0.14981 | Smooth loss: 7.38936

[INFO] Epoch: 23 | Words: 307665126/418195831 | Lr: 0.13215 | Smooth loss: 7.21583

[INFO] Epoch: 24 | Words: 320794675/418195831 | Lr: 0.11645 | Smooth loss: 7.15179

[INFO] Epoch: 25 | Words: 335567580/418195831 | Lr: 0.09879 | Smooth loss: 7.04866

[INFO] Epoch: 26 | Words: 348690406/418195831 | Lr: 0.08310 | Smooth loss: 6.97140

[INFO] Epoch: 27 | Words: 363453014/418195831 | Lr: 0.06545 | Smooth loss: 6.88185

[INFO] Epoch: 28 | Words: 378226646/418195831 | Lr: 0.04779 | Smooth loss: 6.79206

[INFO] Epoch: 29 | Words: 391349910/418195831 | Lr: 0.03210 | Smooth loss: 6.74117

[INFO] Epoch: 30 | Words: 404485041/418195831 | Lr: 0.01639 | Smooth loss: 6.69604

(2) RL Model – Train - Execution Statistics

(First Iteration of Every Epoch – Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cell', name='train_RL_agent', seed=123, gpu='0', epochs=100, batch_size=32, lr=0.0001, max_acts=250, max_path_len=3, gamma=0.99, ent_weight=0.001, act_dropout=0, state_history=1, hidden=[512, 256], debug=0, steps_per_checkpoint=5000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device=device(type='cuda', index=0), dir='./tmp/Amazon_Cellphones/train_RL_agent', checkpoint_dir='./tmp/Amazon_Cellphones/train_RL_agent/checkpoint', log_dir='./tmp/Amazon_Cellphones/train_RL_agent/log')

[INFO] Parameters:['l1.weight', 'l1.bias', 'l2.weight', 'l2.bias', 'actor.weight', 'actor.bias', 'critic.weight', 'critic.bias']

[INFO] epoch/step=1/5000 | loss=56.27084 | ploss=56.27590 | vloss=981446.20370 | entropy=-5.05216 | reward=18.45093

[INFO] Save model to ./tmp/Amazon_Cellphones/train_RL_agent/checkpoint/policy_model_epoch_1.ckpt

[INFO] epoch/step=2/30000 | loss=4.72161 | ploss=4.72594 | vloss=28350626.90854 | entropy=-4.32854 | reward=283.66938

[INFO] Save model to ./tmp/Amazon_Cellphones/train_RL_agent/checkpoint/policy_model_epoch_2.ckpt

[INFO] epoch/step=3/100000 | loss=1.71566 | ploss=1.72059 | vloss=16028139.52008 | entropy=-4.93147 | reward=163.51747

[INFO] Save model to ./tmp/Amazon_Cellphones/train_RL_agent/checkpoint/policy_model_epoch_3.ckpt

[INFO] epoch/step=4/210000 | loss=2.68521 | ploss=2.69027 | vloss=14399656.36550 | entropy=-5.06066 | reward=139.19736

[INFO] Save model to ./tmp/Amazon_Cellphones/train_RL_agent/checkpoint/policy_model_epoch_4.ckpt

[INFO] epoch/step=5/320000 | loss=1.66903 | ploss=1.67410 | vloss=10782385.51766 | entropy=-5.07320 | reward=110.33527

[INFO] Save model to ./tmp/Amazon_Cellphones/train_RL_agent/checkpoint/policy_model_epoch_5.ckpt

(3) RL Model – Test - Execution Statistics

(Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cell', source_name='train_RL_agent', output_folder='test_RL_agent', seed=123, gpu='0', epochs=100, max_acts=250, max_path_len=3, gamma=0.99, state_history=1, hidden=[512, 256], add_products=False, topk=[10, 10, 12], run_path=True, run_eval=True, debug=0, batch_size=512, is_resume_from_checkpoint=0, logging_mode='a', log_file_name='test_agent_log', checkpoint_folder='checkpoint', explainability_score_option=1, run_number='1', is_only_run_specific_epoch=0, device=device(type='cuda', index=0), output_dir='./tmp/Amazon_Cellphones/test_RL_agent')

[INFO] model epoch=1 | count (users)=27879 | ndcg=5.93911 | recall=10.11558 | hit_ratio=14.16475 | precision=1.53739 | invalid_users=7.00000 | execution_timestamp=2023-07-04 21:35:49.560758

[INFO] model epoch=2 | count (users)=27879 | ndcg=5.92715 | recall=10.09293 | hit_ratio=14.17193 | precision=1.54420 | invalid_users=7.00000 | execution_timestamp=2023-07-04 23:42:13.978590

[INFO] model epoch=3 | count (users)=27879 | ndcg=5.97536 | recall=10.11172 | hit_ratio=14.17910 | precision=1.54707 | invalid_users=7.00000 | execution_timestamp=2023-07-05 01:45:54.892499

[INFO] model epoch=4 | count (users)=27879 | ndcg=5.95444 | recall=10.10106 | hit_ratio=14.19704 | precision=1.54887 | invalid_users=7.00000 | execution_timestamp=2023-07-05 03:54:14.072578

[INFO] model epoch=5 | count (users)=27879 | ndcg=5.97054 | recall=10.16329 | hit_ratio=14.22216 | precision=1.55138 | invalid_users=7.00000 | execution_timestamp=2023-07-05 06:00:20.851909

APPENDIX D

EXPERIMENT RESULTS – CLOTHING

(1) KGE Model – Execution Statistics

(First Iteration of Every Epoch)

[INFO] Namespace(dataset='cloth', name='train_transe_model', seed=123, gpu='1', epochs=30, batch_size=64, lr=0.5, weight_decay=0, l2_lambda=0, max_grad_norm=5.0, embed_size=100, num_neg_samples=5, steps_per_checkpoint=10000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device='cpu', dir='./tmp/Amazon_Clothing/train_transe_model', checkpoint_dir='./tmp/Amazon_Clothing/train_transe_model/checkpoint', log_dir='./tmp/Amazon_Clothing/train_transe_model/log')

[INFO] Parameters:['purchase', 'mentions', 'describe_as', 'produced_by', 'belongs_to', 'also_bought', 'also_viewed', 'bought_together', 'user.weight', 'product.weight', 'word.weight', 'related_product.weight', 'brand.weight', 'category.weight', 'purchase_bias.weight', 'mentions_bias.weight', 'describe_as_bias.weight', 'produced_by_bias.weight', 'belongs_to_bias.weight', 'also_bought_bias.weight', 'also_viewed_bias.weight', 'bought_together_bias.weight']

[INFO] Epoch: 01 | Words: 1693584/399498121 | Lr: 0.49788 | Smooth loss: 21.69147

[INFO] Epoch: 02 | Words: 13514211/399498121 | Lr: 0.48309 | Smooth loss: 13.97328

[INFO] Epoch: 03 | Words: 27030792/399498121 | Lr: 0.46617 | Smooth loss: 11.82713

[INFO] Epoch: 04 | Words: 40543202/399498121 | Lr: 0.44926 | Smooth loss: 10.58158

[INFO] Epoch: 05 | Words: 54053045/399498121 | Lr: 0.43235 | Smooth loss: 9.68916

[INFO] Epoch: 06 | Words: 67568430/399498121 | Lr: 0.41543 | Smooth loss: 9.10305

[INFO] Epoch: 07 | Words: 81087305/399498121 | Lr: 0.39851 | Smooth loss: 8.63236

[INFO] Epoch: 08 | Words: 94608288/399498121 | Lr: 0.38159 | Smooth loss: 8.30732

[INFO] Epoch: 09 | Words: 108122969/399498121 | Lr: 0.36468 | Smooth loss: 8.04675

[INFO] Epoch: 10 | Words: 120047043/399498121 | Lr: 0.34975 | Smooth loss: 7.65749

[INFO] Epoch: 11 | Words: 133563624/399498121 | Lr: 0.33284 | Smooth loss: 7.45772

[INFO] Epoch: 12 | Words: 147076034/399498121 | Lr: 0.31592 | Smooth loss: 7.26032

[INFO] Epoch: 13 | Words: 160585877/399498121 | Lr: 0.29902 | Smooth loss: 7.03946

[INFO] Epoch: 14 | Words: 174101262/399498121 | Lr: 0.28210 | Smooth loss: 6.89531

[INFO] Epoch: 15 | Words: 187620137/399498121 | Lr: 0.26518 | Smooth loss: 6.72329

[INFO] Epoch: 16 | Words: 201141120/399498121 | Lr: 0.24826 | Smooth loss: 6.63551

[INFO] Epoch: 17 | Words: 214655801/399498121 | Lr: 0.23134 | Smooth loss: 6.93652

[INFO] Epoch: 18 | Words: 226486781/399498121 | Lr: 0.21654 | Smooth loss: 6.92036

[INFO] Epoch: 19 | Words: 240003969/399498121 | Lr: 0.19962 | Smooth loss: 6.76717

[INFO] Epoch: 20 | Words: 253515256/399498121 | Lr: 0.18271 | Smooth loss: 6.61076

[INFO] Epoch: 21 | Words: 267032378/399498121 | Lr: 0.16579 | Smooth loss: 6.50761

[INFO] Epoch: 22 | Words: 280550059/399498121 | Lr: 0.14887 | Smooth loss: 6.38779

[INFO] Epoch: 23 | Words: 294067416/399498121 | Lr: 0.13195 | Smooth loss: 6.32922

[INFO] Epoch: 24 | Words: 307583878/399498121 | Lr: 0.11504 | Smooth loss: 6.22255

[INFO] Epoch: 25 | Words: 321094157/399498121 | Lr: 0.09813 | Smooth loss: 6.13617

[INFO] Epoch: 26 | Words: 332928617/399498121 | Lr: 0.08332 | Smooth loss: 6.13578

[INFO] Epoch: 27 | Words: 347925288/399498121 | Lr: 0.06455 | Smooth loss: 5.82125

[INFO] Epoch: 28 | Words: 359745915/399498121 | Lr: 0.04975 | Smooth loss: 5.72896

[INFO] Epoch: 29 | Words: 373262496/399498121 | Lr: 0.03284 | Smooth loss: 5.68329

[INFO] Epoch: 30 | Words: 386774906/399498121 | Lr: 0.01592 | Smooth loss: 5.62219

(2) RL Model – Train - Execution Statistics

(First Iteration of Every Epoch – Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cloth', name='train_RL_agent', seed=123, gpu='0', epochs=100, batch_size=32, lr=0.0001, max_acts=250, max_path_len=3, gamma=0.99, ent_weight=0.001, act_dropout=0, state_history=1, hidden=[512, 256], debug=0, steps_per_checkpoint=5000, checkpoint_folder='checkpoint', log_folder='log', log_file_name='train_log.txt', is_resume_from_checkpoint=1, logging_mode='a', device=device(type='cuda', index=0), dir='./tmp/Amazon_Clothing/train_RL_agent',
checkpoint_dir='./tmp/Amazon_Clothing/train_RL_agent/checkpoint',
log_dir='./tmp/Amazon_Clothing/train_RL_agent/log')

[INFO] Parameters:['l1.weight', 'l1.bias', 'l2.weight', 'l2.bias', 'actor.weight', 'actor.bias', 'critic.weight', 'critic.bias']

[INFO] epoch/step=1/5000 | loss=61.40001 | ploss=61.40502 | vloss=1030321.73969 | entropy=-5.00945 | reward=20.13164

[INFO] Save model to ./tmp/Amazon_Clothing/train_agent/checkpoint/policy_model_epoch_1.ckpt

[INFO] epoch/step=2/40000 | loss=3.25712 | ploss=3.26109 | vloss=52848678.69893 | entropy=-3.97390 | reward=536.70443

[INFO] Save model to ./tmp/Amazon_Clothing/train_agent/checkpoint/policy_model_epoch_2.ckpt

[INFO] epoch/step=3/105000 | loss=0.70073 | ploss=0.70519 | vloss=33128534.03819 | entropy=-4.45525 | reward=337.71774

[INFO] Save model to ./tmp/Amazon_Clothing/train_agent/checkpoint/policy_model_epoch_3.ckpt

[INFO] epoch/step=4/225000 | loss=0.98947 | ploss=0.99444 | vloss=14879036.77535 | entropy=-4.96894 | reward=152.13498

[INFO] Save model to ./tmp/Amazon_Clothing/train_agent/checkpoint/policy_model_epoch_4.ckpt

[INFO] epoch/step=5/370000 | loss=1.01398 | ploss=1.01899 | vloss=17741003.27381 | entropy=-5.01831 | reward=180.20049

[INFO] Save model to ./tmp/Amazon_Clothing/train_agent/checkpoint/policy_model_epoch_5.ckpt

(3) RL Model – Test - Execution Statistics

(Showing Only the First 5 Epochs Details)

[INFO] Namespace(dataset='cloth', source_name='train_RL_agent', output_folder='test_RL_agent', seed=123, gpu='0', epochs=100, max_acts=250, max_path_len=3, gamma=0.99, state_history=1, hidden=[512, 256], add_products=False, topk=[10, 10, 12], run_path=True, run_eval=True, debug=0, batch_size=512, is_resume_from_checkpoint=0, logging_mode='a', log_file_name='test_agent_log', checkpoint_folder='checkpoint', explainability_score_option=1, run_number='1', is_only_run_specific_epoch=0, device=device(type='cuda', index=0), output_dir='./tmp/Amazon_Clothing/test_RL_agent')

[INFO] model epoch=1 | count (users)=39387 | ndcg=3.25963 | recall=5.60079 | hit_ratio=8.08917 | precision=0.84624 | invalid_users=1.00000 | execution_timestamp=2023-07-04 20:54:09.247738

[INFO] model epoch=2 | count (users)=39387 | ndcg=3.26301 | recall=5.59971 | hit_ratio=8.14503 | precision=0.85360 | invalid_users=1.00000 | execution_timestamp=2023-07-04 22:42:12.656890

[INFO] model epoch=3 | count (users)=39387 | ndcg=3.27801 | recall=5.64154 | hit_ratio=8.16026 | precision=0.85462 | invalid_users=1.00000 | execution_timestamp=2023-07-05 00:29:52.179086

[INFO] model epoch=4 | count (users)=39387 | ndcg=3.24676 | recall=5.57934 | hit_ratio=8.06886 | precision=0.84548 | invalid_users=1.00000 | execution_timestamp=2023-07-05 02:14:01.424329

[INFO] model epoch=5 | count (users)=39387 | ndcg=3.29899 | recall=5.64402 | hit_ratio=8.16788 | precision=0.85487 | invalid_users=1.00000 | execution_timestamp=2023-07-05 04:05:34.255887