

Forgetting in Knowledge Graph Based Recommender Systems

Xu Wang¹ ^a and Christopher Brewster^{1,2} ^b

¹*Institute of Data Science, Maastricht University, Paul-Henri Spaaklaan 1, 6229 GT, Maastricht, The Netherlands*

²*Data Science Group, TNO, Kampweg, Soesterberg, The Netherlands*

Keywords: Recommender System, Knowledge Graph, Forgetting, Datalog.

Abstract: Recommender systems need to contend with continuous changes in both search spaces and user profiles. The set of items in the search space is usually treated as continuously expanding, however, users also purchase items or change their requirements. This raises the issue of how to “forget” an item after purchase or consumption. This paper addresses the issue of “forgetting” in knowledge graph-based recommender systems. We propose an innovative method for identifying and removing unnecessary or irrelevant triples from the graph itself. Using this approach, we simplify the knowledge graph while maintaining the quality of the recommendations. We also introduce several metrics to assess the impact of forgetting in knowledge graph-based recommender systems. Our experiments demonstrate that incorporating consideration of impact in the forgetting process can enhance the efficiency of the recommender system without compromising the quality of its recommendations.

1 INTRODUCTION

Recommender systems function as a specialized subset within decision-making or information filtering frameworks and are widely used across diverse applications/platforms (Roy and Dutta, 2022; Kreutz and Schenkel, 2022). Platforms like Netflix¹ and Amazon² use these systems to predict and recommend content tailored to user preferences. Typically, these systems are categorized into three or four distinct types: collaborative filtering, content-based, knowledge-based (occasionally regarded as a derivative of content-based), and hybrid recommender systems (Lu et al., 2015).


Knowledge graph-based recommender systems use domain knowledge for decision making and form an important category of system as they can be used to address the cold start problem. Unlike collaborative filtering, which operates on the premise “other similar users also liked this,” and content-based systems, which suggest “you might be interested in the content of this”, knowledge graph-based systems can be summarized as “based on certain attributes of yours, it is inferred that you would like this.” In other words, knowledge graph-based recommender systems use rules, reasoning, or constraints applied to domain knowledge to

make recommendation decisions (Le et al., 2023; Ai et al., 2018).

The concept of forgetting has been a part of computer science and artificial intelligence research for at least 30 years. Following Lin and Reiter in 1995 with the introduction of the “Forget” concept in artificial intelligence (Lin and Reiter, 1994), a plethora of studies focusing on forgetting in AI research have since emerged. In the realm of knowledge representation and reasoning, the theoretical study of forgetting has been prominently featured in numerous recent research papers (Eiter and Kern-Isberner, 2018; Delgrande, 2017).

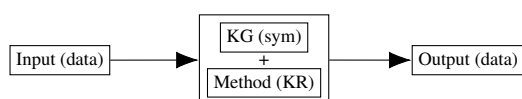
In this paper, we aim to integrate the idea of “forgetting” into knowledge graph-based recommender systems (KG-based recommender systems) in order to adapt to the dynamic changing needs of a user. The “forgetting” used in this paper is equivalent to “becoming unaware” in (van Ditmarsch et al., 2008). The knowledge graph-based recommender system we considered in this paper will be one with path-based methods or unified methods as described in (Guo et al., 2022). In other words, this kind of recommender system uses the knowledge graph as background knowledge and applies some knowledge-based methods (including machine learning methods and/or symbolic reasoning methods) to make recommendation decisions. The following is a boxology representation (boxology notation is introduced in (Harmelen and Teije, 2019)) of this kind of recommender system:

^a  <https://orcid.org/0000-0002-7585-759X>

^b  <https://orcid.org/0000-0001-6594-9178>

¹ <https://www.netflix.com/>

² <https://www.amazon.com/>



where **Input (data)** **Output (data)** means "model-free" input and output data of recommender system; **KG (sym)** means the "model-based" knowledge graph data; **Method (KR)** means some forms of deductive inference, which would be a rule based reasoning method over a knowledge graph (Ma et al., 2019; Balloccu et al., 2022) or the symbolic explanation of some machine learning based method.

The main contribution of this paper is to provide:

- 1) a function for identifying facts (triples) to forget from a knowledge graph-based recommender system;
- 2) a novel task in the forgetting of facts (triples) in knowledge graph-based recommender systems;
- 3) a number of metrics to measure the impact of forgetting in knowledge graph-based recommender systems.

In this paper, we will mainly focus on three research questions: 1. What is forgetting in KG-based recommender system? 2. How to perform forgetting in KG-based recommender system? 3. How to quantitatively measure the impact of forgetting?

2 RELATED WORK

Recent research in recommender systems has incorporated knowledge graphs and ontologies. Tarus et al. (Tarus et al., 2017) develop a hybrid system for e-learning that combines an ontology with sequential pattern mining to enhance personalization and address challenges like cold-start. Carrer-Neto et al. (Carrer-Neto et al., 2012) propose a hybrid movie recommendation system using knowledge-based techniques and social data, guided by Semantic Web principles. Dong et al. (Dong et al., 2020) introduce an interactive fashion design recommender that merges sensory evaluation, fuzzy logic, and an ontology-based knowledge base. Brisse et al. (Brisse et al., 2022) present KRAKEN, a knowledge-based system for security analysis, utilizing a knowledge base of adversarial tactics with a visual tool. Esheiba et al. (Esheiba et al., 2021) offer a hybrid system for selecting Product-Service Systems, using ontologies, constraint satisfaction, and utility functions for customer alignment. Arnaoutaki et al. (Arnaoutaki et al., 2019) describe a recommender for Mobility as a Service plans, combining Constraint Satisfaction Problem solving with weighted similarity.

There is some existing research which considers forgetting in recommender system. Jin et. al. (Jin

et al., 2023) introduce PMORS, a recommender system incorporating the Ebbinghaus Forgetting Curve for recent negative feedback, optimized for video platforms and showing improved results on WeChat Channels. Liu et. al. (Liu et al., 2022) propose AltEraser, an unlearning technique for neural recommendation models, focusing on efficiency and effectiveness with a warm-start strategy and second-order optimization. Zhang et. al. (Zhang and Lu, 2020) present the MTMF model for temporal recommender systems, combining a personalized time weight and item transition matrix, leading to superior accuracy on MovieLens. Matuszyk et. al. (Matuszyk et al., 2015) explore new forgetting techniques in incremental matrix factorization for recommender systems, enhancing predictive accuracy and confirming the benefits of strategic forgetting. Verachtert et. al. (Verachtert et al., 2022) emphasize the importance of an optimal training window size in recommender systems, showing improved recommendation quality with recent data usage. Matuszyk et. al. (Matuszyk et al., 2017) introduce unsupervised forgetting techniques with algorithms for recommender systems, significantly enhancing accuracy across datasets. Tavakolian et. al. (Tavakolian et al., 2012) develop WmIDForg, a recommender system using web and content mining with a forgetting mechanism, improving precision on the EachMovie dataset. Li et. al. (Li et al., 2024) address unlearning in recommender systems with the LASER framework, enabling efficient data deletion while maintaining model utility, validated with real-world datasets.

3 MOTIVATION

Online streaming platforms like YouTube and Netflix enable users to flag content as uninteresting through options like 'Not Interested' or 'Do Not Recommend This'. This feedback leads to updates in the platform's recommendation algorithms, aligning them more closely with user preferences by filtering out content that users have indicated they don't like or have already seen. In knowledge-based recommender systems, such updates are crucial as they involve "forgetting" outdated information to keep pace with changing user demands.

The introduction of the concept of knowledge forgetting in recommender system is intended to make the recommendation process more efficient and adaptive. Motivations to introduce a forgetting capability into recommender systems include:

- **Reducing Computational Complexity:** Recommender systems can enhance efficiency and conserve computational resources by regularly removing outdated or irrelevant data, par-

ticularly in fast-paced sectors like news or fashion where current information is essential. • **Products No Longer Available on the Server:** In e-commerce, regularly updating product lines and removing discontinued items from the recommendation pool saves storage and maintains high-quality, relevant suggestions, enhancing customer satisfaction and increasing conversion rates. • **Changing the Knowledge Graph Rather Than User Profiles (due to product risks, technological updates, or changing trends):** A Knowledge Graph (KG) in recommendation systems needs timely updates when products become unsafe or unpopular due to safety risks, technological changes, or trends, ensuring accurate recommendations and preventing the negative impact on overall user profiles from specific product issues. • **Legal Implications:** Recommendation systems must quickly delete certain data, such as user information due to privacy laws or details about banned or restricted products, to comply with legal changes, protect users and the platform from legal risks, and maintain user trust. Applying forgetting to knowledge-based recommender system is a novel and unusual research topic in the domain of recommender system research. We expect this approach will be beneficial by removing irrelevant knowledge/information, and optimising the knowledge-based recommender system to the dynamic user preferences or needs.

4 PRELIMINARIES

In this section, we introduce the foundational concepts and notations that will be used throughout. We denote by Γ the set of all rules, Φ the set of all triples, and Ψ the set of all atoms.

Definition 1 (Datalog Rule). *A Datalog rule is an expression composed of positive atoms $\Psi_P \subset \Psi$. It takes the form of a Horn clause, which is an implication where the consequent (head) is a single atom and the antecedent (body) is a conjunction of atoms:*

$$\text{head} \leftarrow \text{body}_1 \wedge \text{body}_2 \wedge \dots \wedge \text{body}_n$$

Here, each of $\text{head}, \text{body}_1, \text{body}_2, \dots, \text{body}_n$ is an atom element of Ψ_P . Datalog rules are characterized by having only positive literals in both the head and body of the rule, reflective of their origin in logic programming.

Definition 2 (Least Model). *Given a set of Datalog rules $R \in \Gamma$. Given a set of atom $F \in \Phi$ to represent the ground-true facts. For any set of atom M , M is the model of $F \cup R$ if and only if $F \cup R \models M$. For one model M' of $F \cup R$, M' is the Least Model of $F \cup R$ if and only if for any other model M'' of $F \cup R$, $M' \prec M''$ where*

\prec means subsume which is the more-general-relation between models.

The concept of a recommender system is formalized building upon the definition provided by (Adomavicius and Tuzhilin, 2005).

Definition 3 (Recommender System). *Let U be the set of all users, and I the set of all items that can be recommended. Let R be the set of non-negative real numbers representing utility values. The utility function $f : U \times I \rightarrow R$ evaluates the usefulness of item $i \in I$ to user $u \in U$. A recommender system aims to recommend an item $i \in I$ to a user $u \in U$ such that $f(u, i) \geq f(u, i')$ for any other item $i' \in I$, effectively maximizing the perceived utility for the user.*

To facilitate the integration of knowledge graphs into logic-based systems, we introduce an atomic representation for triples:

Definition 4 (Atomic Representation of Triples). *For each triple $t \in \Phi$, represented as $t = \langle s, p, o \rangle$ where s , p , and o correspond to the subject, predicate, and object of t , respectively, we define the atomic formula representation $af(t)$ as follows:*

$$af(t) = p'(s', o')$$

Here, p' is a predicate symbol representing the relationship p , and s' and o' are the corresponding arguments for the subject s and object o within the logical framework. This representation enables the direct use of triples in logical inference processes, aligning the structure of knowledge graphs with formal reasoning mechanisms.

5 METHODOLOGY

In this section, we introduce a comprehensive methodology for identifying forgettable triples in a knowledge graph, detailing the process of forgetting these triples and proposing methods to measure the impact of such forgetting. Assuming Γ is a set of all Datalog rules, Φ a set of all triples, Ψ a set of all atoms in this section.

5.1 Forgetting Task in Knowledge Graph Based Recommender System

A forgetting task involves selectively removing certain triples from the knowledge graph to better match a user's changing needs.

Definition 5 (Forgetting Task in KG-based Recommender System). *Let G denote a knowledge graph that comprises a set of triples $T \subseteq \Phi$. Suppose RS is a recommender system based on the knowledge graph G ,*

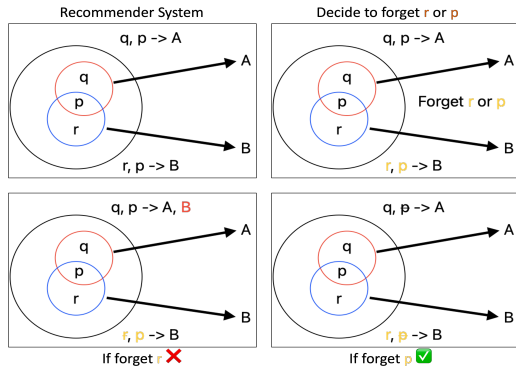


Figure 1: Example of forgetting. **Top Left:** The KG consists of elements $q, p,$ and $r,$ and there are two rules in KG-based recommender system: one that implies A from q and p ($q, p \rightarrow A$) and another that implies B from r and p ($r, p \rightarrow B$). **Top Right:** We decided to forget r or $p.$ **Bottom Left:** Attempts to demonstrate the forgetting of element $r.$ However, the rule ($r, p \rightarrow B$) is now problematic. If r is forgotten, the rule simplifies to $p \rightarrow B,$ which would affect the other rule ($q, p \rightarrow A$), effectively transforming it into ($q, p \rightarrow A, B$) because p alone would now trigger both A and $B.$ This is not the desired outcome, as it conflates the conditions for recommending A and $B.$ **Bottom Right:** Attempts to demonstrate the forgetting of element $p.$

incorporating a set of Datalog rules $R \subseteq \Gamma$ which dictate the recommendation logic of RS. Consider $\langle I, O \rangle$ as the input-output pair for RS, where $I \subseteq \Phi$ represents the inputs and $O \subseteq \Phi$ represents the outputs such that the intersection $I \cap T \cap R$ logically entails $O.$ A forgetting task is then defined as a transformation function $forget : \Phi \times \Gamma \rightarrow \Phi \times \Gamma$ which accepts a set of triples T and a set of Datalog rules $R,$ and yields a modified set of triples $T' \subseteq T$ and a revised set of Datalog rules $R',$ ensuring that for any input $I,$ the new system RS' with its updated knowledge graph $G' (T')$ and recommendation mechanism representation R' provides the same recommendations O such that $I \cap T' \cap R' \models O$ if and only if $I \cap T \cap R \models O,$ excluding any elements related to the forgotten aspects.

Figure 1 exemplifies the challenges in a forgetting task where removing an element from a KG-based recommender system can lead to unintended consequences in the recommendation logic, as seen when trying to forget 'r', which disrupts the distinct recommendation conditions for 'A' and 'B'.

A recommendation task in KG-based recommender system can be considered as a "link-prediction" task, where we try to predict the missing recommendation relations between user and item. This means that the output O of recommender system RS is not in the background knowledge graph G or $G'.$ Then we have a proposition about this.

Proposition 1. Let G denote a knowledge graph that

comprises a set of triples $T \subseteq \Phi.$ Suppose RS is a recommender system based on the knowledge graph $G,$ incorporating a set of Datalog rules $R \subseteq \Gamma$ which dictate the recommendation logic of RS. Consider $\langle I, O \rangle$ as the input-output pair for RS, where $I \subseteq T$ represents the inputs and $O \subseteq \Phi$ represents the outputs such that the intersection $I \cap T \cap R$ logically entails $O.$ We have that $\forall t \in O, t \notin T.$

The input-output pair $\langle I, O \rangle$ follows the Least Model semantics according to the recommendation mechanism representation R or R' in the KG-based recommender system. Then we have a proposition about the input-output pair $\langle I, O \rangle.$

Proposition 2. Let G denote a knowledge graph that comprises a set of triples $T \subseteq \Phi.$ Suppose RS is a recommender system based on the knowledge graph $G,$ incorporating a set of Datalog rules $R \subseteq \Gamma$ which dictate the recommendation logic of RS. Consider $\langle I, O \rangle$ as the input-output pair for RS, where $I \subseteq \Phi$ represents the inputs and $O \subseteq \Phi$ represents the outputs such that the intersection $I \cap T \cap R$ logically entails $O.$ Then we have that $I \cup R$ has the Least Model $O.$ This means that with input $I,$ recommender system RS could have least output O as recommendation results. For the set of all outputs of RS, denoted as $O_s,$ we have that $T \cup R$ has the Least Model $O_s,$ such that RS always output O_s with background knowledge graph G and recommendation logic $R.$

5.2 Passive and Intentional Forgetting

Passive forgetting in the context of a KG-based recommender system refers to the process of omitting certain knowledge due to changes in user needs or external factors. Unlike other forms of forgetting, passive forgetting is reactive, occurring in response to shifts in the external environment, particularly the preferences and requirements of users.

Definition 6 (Passive Forgetting in KG-based Recommender System). Let RS be a KG-based recommender system, utilizing a background knowledge graph KG with triples $T \subseteq \Phi,$ and a set of Datalog rules R representing the recommendation logic of RS. Define the search space as the subset of triples encompassing all outputs no longer aligning with the updated user needs and any triples that infer these outputs via $R.$ Passive forgetting is then defined as a transformation function $PF : \Phi \times \Gamma \times \Phi \times \Phi \rightarrow \Phi,$ where a change in user needs from O to O' prompts PF to produce a refined set of triples $T' \subseteq T$ by excluding elements from $T.$ This ensures logical consistency in recommendations, where $T \cup R \models O$ if and only if $T' \cup R \models O',$ effectively adapting the knowledge graph to the updated preferences.

In contrast to passive forgetting, intentional forgetting is a proactive approach that considers the entirety of the background knowledge graph, rather than just the parts related to immediate user needs. This form of forgetting is about optimizing the knowledge graph by removing information that is deemed irrelevant or unnecessary for the current state and goals of the recommender system.

Definition 7 (Intentional Forgetting in KG-based Recommender System). *Let RS be a KG-based recommender system, utilizing a background knowledge graph KG with triples $T \subseteq \Phi$, and a set of Datalog rules R representing the recommendation logic of RS . Intentional forgetting is characterized as an optimization function $IF : \Phi \times \Gamma \rightarrow \Phi$, which processes T and R to yield a streamlined set of triples $T'' \subseteq T$. This operation is designed to preserve the recommendation capabilities of the system, such that for the set of all outputs O of RS , $T'' \cup R \models O$ if and only if $T \cup R \models O$. The purpose of this function is to enhance the efficiency and relevancy of the knowledge graph by deliberately discarding superfluous or outdated information.*

5.3 Forgetting Triple

We use the "fires" definition inspired from (Betz et al., 2022) for rules. Informally, a rule "fires" means that the triples in the body of this rule should be covered by knowledge graph.

Definition 8 (Rule "Fires"). *Let $r \in \Gamma$ be a Datalog rule. Let G be a knowledge graph that contains a set of triples $T \subseteq \Phi$. r fires w.r.t. (with respect to) G iff the body b of r satisfies $b \subseteq T$. We also call r the fired rule w.r.t. graph G .*

The subsequent definitions and propositions consider only the rules that "fire".

We now discuss how to identify triples suitable for forgetting. Our concept of forgetting is motivated by (Lin and Reiter, 1994), which discusses forgetting facts or relations. The first step is to determine which triples in the knowledge graph can be forgotten. The aim of forgetting is to simplify knowledge while retaining essential capabilities, akin to Occam's Razor: "Keep things simple".

Definition 9 (Search Space for Passive Forgetting). *Let G be a knowledge graph that contains a set of triples $T \subseteq \Phi$. Let $R \subseteq \Gamma$ be a set of fired rules w.r.t. G . Let RS be a KG-based recommender system, with background knowledge G and recommendation logic representation R . Let $OT \subseteq T$ be a set of triples that would be forgotten in the output of RS . Then the search space for passive forgetting in RS w.r.t. OT is $SSPas_{RS}(OT) = \{t | t \in \text{body}(r) \cup$*

$\text{head}(r) \cup T, \text{head}(r) \in OT, \text{ where } r \in R\}$. We also call the search space $SSPas_{RS}(OT)$ as the forgettable triple set in G w.r.t. OT .

Differ from passive forgetting, **search space for intentional forgetting** $SSInt_{RS}$ is the background knowledge graph G of the KG-based recommender system RS , or we can say that the forgettable triple set of intentional forgetting would be all the triples in the background knowledge graph.

Following is a proposition related to the intersectionality of the search space for passive forgetting:

Proposition 3. *Let G be a knowledge graph that contains a set of triples $T \subseteq \Phi$. Let $OT1, OT2 \in T$ be two triple sets that would be forgotten in the output of RS . We have $SSPas_{RS}(OT1 \cap OT2) = SSPas_{RS}(OT1) \cap SSPas_{RS}(OT2)$.*

Having constructed the search space, we can optimize the rules. As we mentioned above, forgetting in this paper is equivalent to "becoming unaware", which means that we will no longer consider the truth of a triple when we decide to forget this triple. Then, we can optimise the Datalog rules by ignoring the forgetting triples. The optimisation of a rule r with forgetting involves creating an optimized rule r' by excluding forgettable triples from the body b of r , while keeping the head h unchanged. This process refines the rule by removing redundant elements without altering its logical impact.

Definition 10 (Rule optimisation with Forgetting). *Let G be a knowledge graph that contains a set of triples $T \subseteq \Phi$. Let $r \in R$ is a fired Datalog rule w.r.t. G . Let RS be a KG-based recommender system. Let t be the one triple in search space for (passive or intentional) forgetting in RS . If $t \neq h$ and $t \in b$, we have rule $r' : h \leftarrow b \setminus \{t\}$ is the optimised rule of r w.r.t. G with forgetting t , where h is the head of r and b is the body of r .*

This optimization ensures the rule still fires as it only removes triples from the rule's body.

Proposition 4. *Let G be a knowledge graph that contains a set of triples $T \subseteq \Phi$. Let $r \in \Gamma$ be a fired rule w.r.t. G . If r' is the optimised rule of r w.r.t. G with forgetting, we have that r' is the fired rule w.r.t. G .*

5.4 Quantitative Measures of Impact

In the realm of KG-based recommender systems, the act of forgetting specific triples can have a profound impact on various aspects of system performance and behavior. To effectively quantify and understand these impacts, we introduce two distinct metrics: impact on Least Model, and impact on weakest sufficient conditions. Assuming G a knowledge graph that contains

a set of triples $T \subseteq \Phi$, $R \subseteq \Gamma$ a set of fired Datalog rules w.r.t. G , RS a KG-based recommender system that has background G and recommendation logic representation R , and SS the search space for (passive or intentional) forgetting in RS .

Least Model represents consistent and complete sets of conclusions that can be drawn from a knowledge graph under a given set of rules. Assessing the impact on Least Model is essential for understanding how the omission of certain triples affects the integrity and coherence of the system's logical foundations.

$$Impact_{LM} = 1 - \frac{|Least Model after Forgetting|}{|Least Model before Forgetting|} \quad (1)$$

The impact on weakest sufficient conditions (WSC) is evaluated by comparing the normalized scores of paths within the knowledge graph before and after the forgetting of a specific triple. We use the Personalized Page Rank (PPR) algorithm for the personalized importance of entities relative to a target entity, and the eigenvector centrality algorithm for the global importance of entities (Schoenberg, 1969). Then we have the entity importance:

$$Imp(e) = \alpha * eigenvector(e) + \beta * PPR(e) - \tau * (eigenvector(e) * PPR(e)) \quad (2)$$

where $eigenvector(e)$ is the eigenvector centrality of entity e and $PPR(e)$ is the personalized page rank score of entity e . α and β are the weight of eigenvector centrality and PPR. τ serves as a damping factor to reduce the score of nodes that are highly central both globally and from the perspective of the target node, addressing potential redundancy.

For predicate in triples, we use the degree centrality to compute the importance of predicates. Then, we have the WSC impact of forgetting one triples in graph:

$$Impact_{WSC}(t) = \omega_{predicate} * degree(p) + \omega_{entity} * Imp(s) + \omega_{entity} * Imp(o) \quad (3)$$

where $\omega_{predicate}$ and ω_{entity} are the weight of importance score of predicate p and entity. $degree(p)$ is the degree centrality of predicate p . s, p, o are the subject, predicate and object of triple t respectively.

6 EXPERIMENTS AND RESULTS

6.1 Dataset

We use the dataset from the paper of Balloccu et. al. (Balloccu et al., 2022). Balloccu et. al. introduce

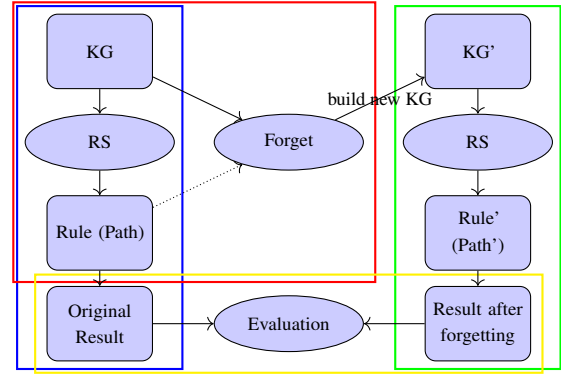


Figure 2: Overview of experiment setup. **Step1 (Reproduce)**: Running recommendation experiments with original knowledge graph and datasets. **Step2 (Forget)**: Forgetting based on original recommendation process to build new KG. **Step3 (Rerun)**: Using new-built KG to rerun recommendation experiments. **Step4 (Evaluation)**: Evaluating the results of original experiments and rerun experiments.

an innovative path-based recommendation approach that leverages user-item interaction paths to generate more transparent and interpretable recommendations, focusing on the significance of the path's context and its contribution to enhancing the explainability of the recommendations. They provide the knowledge graph which is used for path-based recommendations and also the path explanation for recommendation, where the path explanations could be considered as the rule in our paper.

6.2 Experiment Setup

We conduct experiments on forgetting using existing recommendation systems based on knowledge graphs and paths (where paths can be considered as logical rules). The experiment is divided into several parts: 1) Reproducing recommendations using the original knowledge graph and paths from experiments of (Balloccu et al., 2022), 2) Based on the reproduced recommendation results and paths, two kinds (with considering impact of Least Model or Weakest Sufficient Condition) of intentional forgetting are used to construct a new knowledge graph, 3) Re-experimenting using the new knowledge graph and measuring the impact of forgetting using two different methods.

In Figure 2, we illustrate the experimental setup employed in our study, which is structured into four distinct steps: 1. **Reproduce**: Initially, we conduct recommendation experiments using the original Knowledge Graph (KG) and datasets (highlighted in blue). The recommendation system (RS) utilizes the KG to derive rules (paths) for generating original results. 2. **Forget**: Subsequently, we implement a 'forgetting' process based on what we introduced in Methodology

section (highlighted in red) that modifies the original KG based on original recommendation. This step involves selectively forgetting certain triples to refine the KG, leading to the construction of a new knowledge graph (KG'). 3. **Rerun:** With the newly constructed KG', we rerun the recommendation experiments (highlighted in green). The updated KG' provides a revised set of rules (paths') that the RS employs to generate new results after forgetting. 4. **Evaluation:** Finally, we evaluate the outcomes of the recommendation processes (highlighted in yellow) by comparing the original results with the results obtained after the forgetting process. This evaluation aims to assess the impact of the forgetting process on the quality and relevance of the recommendations. This experimental framework allows us to systematically explore the effects of information forgetting on recommendation systems and to understand how the selective omission of data influences the generation of recommendations.

Two forms of forgetting are applied in the experiments: intentional forgetting influenced by the Least Model and intentional forgetting influenced by the Weakest Sufficient Condition (WSC). When compared individually to WSC, forgetting influenced by the Least Model can be regarded as a form of active forgetting impacted by the Strongest Necessary Condition, as it considers the intuitive effects on the recommendation outcomes. For the Forgetting with Least Model approach, we assess whether the candidate forgetting triplets are present in the Least Model of the input-output process of recommendation. In the case of Forgetting with WSC, we calculate the WSC score for each candidate forgetting triplet, then select and retain the top 95% of triplets by descending order, thus constructing a new knowledge graph. The code of our experiments are open access³.

6.3 Evaluation

Our evaluation method investigates the impact of alterations to different knowledge graphs on a recommendation system, employing metrics such as NDCG, Hit Ratio (HR), recall, precision, Linking Interaction Recency (LIR), Shared Entity Popularity (SEP), and Explanation Type Diversity (ETD) (Wang et al., 2013; Gunawardana and Shani, 2015). NDCG assesses ranking quality by valuing highly relevant items at the top. HR reflects the occurrence of relevant items in recommendations. Recall measures the system's capacity to identify all relevant items, whereas precision evaluates the accuracy of positive predictions. LIR explores how recent interactions between users and items affect recommendations. SEP considers the influence

³https://github.com/XuWangDACS/Forget_KGRS

of item popularity among users on recommendation quality. ETD measures the diversity of explanations for recommendations, aiming to improve user trust and satisfaction.

6.4 Results

Table 1 provides an in-depth analysis of the impact of various forgetting methods and optimization strategies on recommendation system performance, assessed across a range of metrics such as NDCG, HR, recall, precision, LIR, SEP, and ETD. These metrics evaluate the recommendation system's ranking quality, hit rate, recall, precision, long-tail item recommendation efficacy, sequence preference, and recommendation diversity.

Our primary focus is on the degradation of the recommendation results (M, W) after forgetting, compared to the original recommendation results (O), under the same optimization parameters (opt). Analysis of Table 1 reveals that the maximum reduction is less than 0.01.

Key findings include: 1. *Performance Stability:* Compared to traditional methods, the forgetting approaches Least Model and Weakest Sufficient Condition consistently impact recommendation system performance across various strategies, notably in NDCG and HR metrics, demonstrating their ability to effectively modify the knowledge graph without reducing recommendation quality. 2. *Optimal Configurations:* Optimal performance under each setup is denoted by bold figures within the table. For instance, the EM configuration excels in the NDCG metric, suggesting that the Least Model forgetting approach, when paired with ETD optimization, significantly enhances recommendation list quality. This pattern recurs across different metrics and configurations, facilitating the identification of the most influential forgetting strategies and optimization combinations for specific performance indicators. 3. *Forgetting Methods' Efficacy:* Despite the informational reduction by the forgetting methods, there is no notable decline in the recommendation systems' performance, indicative of a deliberate balance in forgetting strategy design that preserves the recommendation systems' core functionalities and performance. This equilibrium suggests that carefully crafted forgetting methods can modify and optimize the knowledge graph while preserving or even improving recommendation quality.

In essence, this table not only details the specific impacts of diverse forgetting methods and optimization strategies on recommendation system performance but also accentuates the forgetting methods' efficacy and stability in maintaining or enhancing system perfor-

Table 1: Average score of each metrics for each opt. 'E', 'S', and 'L' stand for optimization of ETD, SEP, and LIR respectively, while 'O', 'M', and 'W' denote Original recommendation, recommendation of forgetting with Least Model, and recommendation of forgetting with Weakest Sufficient Condition.

Metric	EO	EM	EW	SO	SM	SW	LO	LM	LW	ELO	ELM	ELW	SLO	SLM	SLW	ESO	ESM	ESW	ESLO	ESLM	ESLW
NDCG	0.081	0.083	0.079	0.097	0.097	0.097	0.088	0.085	0.086	0.086	0.084	0.082	0.093	0.092	0.092	0.095	0.095	0.094	0.094	0.092	0.091
HR	0.152	0.156	0.148	0.184	0.186	0.185	0.166	0.165	0.163	0.161	0.157	0.153	0.176	0.176	0.174	0.173	0.173	0.171	0.174	0.169	0.168
recall	0.006	0.006	0.006	0.008	0.008	0.008	0.007	0.006	0.006	0.007	0.006	0.006	0.008	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007
precision	0.018	0.019	0.017	0.023	0.023	0.023	0.020	0.020	0.020	0.019	0.019	0.018	0.022	0.022	0.021	0.021	0.021	0.021	0.021	0.021	0.021
LIR	0.149	0.151	0.149	0.138	0.141	0.141	0.357	0.359	0.359	0.345	0.346	0.346	0.353	0.354	0.354	0.137	0.137	0.137	0.340	0.340	0.341
SEP	0.613	0.612	0.612	0.927	0.927	0.927	0.588	0.587	0.587	0.655	0.655	0.655	0.881	0.881	0.881	0.884	0.883	0.884	0.853	0.853	0.853
ETD	0.385	0.383	0.383	0.198	0.198	0.198	0.129	0.129	0.130	0.390	0.388	0.390	0.165	0.165	0.165	0.396	0.395	0.396	0.369	0.369	0.370

mance. These insights are invaluable for future research, highlighting the potential for designing forgetting strategies that flexibly adjust and optimize knowledge without compromising key performance metrics.

7 CONCLUSION AND DISCUSSION

This study successfully integrates knowledge forgetting mechanisms into recommendation systems, demonstrating through empirical validation that the selective removal of triples does not compromise recommendation quality. Our findings underscore the resilience and efficacy of the Least Model and Weakest Sufficient Condition forgetting methods, which adeptly adjust the knowledge graph while maintaining system performance. The research confirms that sophisticated forgetting strategies can enhance system efficiency without impacting the core functionalities of recommendation systems.

Looking ahead, the potential for expanding the scope of forgetting to include more complex elements like predicates presents a significant opportunity for further research. Future studies should also explore broader metrics for measuring the impact of forgetting and extend the applicability of these techniques to a wider range of recommendation systems. By addressing these areas, research can develop innovative strategies that adapt to the evolving preferences of users and the dynamic nature of knowledge, thereby advancing the field of recommendation systems.

ACKNOWLEDGEMENT

This work is funded by HORIZON EUROPE project "EU-FarmBook: supporting knowledge exchange between all AKIS actors in the European Union" (Grant ID: 101060382).

REFERENCES

Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749.

Ai, Q., Azizi, V., Chen, X., and Zhang, Y. (2018). Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms*, 11(9):137.

Arnaoutaki, K., Magoutas, B., Bothos, E., and Mentzas, G. (2019). A hybrid knowledge-based recommender for mobility-as-a-service. In *Proceedings of the 16th International Joint Conference on e-Business and Telecommunications*. SCITEPRESS - Science and Technology Publications.

Balocco, G., Boratto, L., Fenu, G., and Marras, M. (2022). Post processing recommender systems with knowledge graphs for recency, popularity, and diversity of explanations. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22*, page 646–656, New York, NY, USA. Association for Computing Machinery.

Betz, P., Meilicke, C., and Stuckenschmidt, H. (2022). Adversarial explanations for knowledge graph embeddings. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-2022*. International Joint Conferences on Artificial Intelligence Organization.

Brisse, R., Boche, S., Majorczyk, F., and Lalande, J.-F. (2022). *KRAKEN: A Knowledge-Based Recommender System for Analysts, to Kick Exploration up a Notch*, page 1–17. Springer International Publishing.

Carrer-Neto, W., Hernández-Alcaraz, M. L., Valencia-García, R., and García-Sánchez, F. (2012). Social knowledge-based recommender system. application to the movies domain. *Expert Systems with Applications*, 39(12):10990–11000.

Delgrande, J. P. (2017). A knowledge level account of forgetting. *Journal of Artificial Intelligence Research*, 60:1165–1213.

Dong, M., Zeng, X., Koehl, L., and Zhang, J. (2020). An interactive knowledge-based recommender system for fashion product design in the big data environment. *Information Sciences*, 540:469–488.

Eiter, T. and Kern-Isberner, G. (2018). A brief survey on forgetting from a knowledge representation and reasoning perspective. *KI - Künstliche Intelligenz*, 33(1):9–33.

- Esheiba, L., Elgammal, A., Helal, I. M. A., and El-Sharkawi, M. E. (2021). A hybrid knowledge-based recommender for product-service systems mass customization. *Information*, 12(8):296.
- Gunawardana, A. and Shani, G. (2015). *Evaluating Recommender Systems*, page 265–308. Springer US.
- Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2022). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3549–3568.
- Harmelen, F. v. and Teije, A. t. (2019). A boxology of design patterns for hybrid learning and reasoning systems. *Journal of Web Engineering*, 18(1):97–124.
- Jin, J., Zhang, Z., Li, Z., Gao, X., Yang, X., Xiao, L., and Jiang, J. (2023). Pareto-based multi-objective recommender system with forgetting curve.
- Kreutz, C. K. and Schenkel, R. (2022). Scientific paper recommendation systems: a literature review of recent publications. *International Journal on Digital Libraries*, 23(4):335–369.
- Le, N. L., Abel, M.-H., and Gouspillou, P. (2023). A constraint-based recommender system via rdf knowledge graphs. In *2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE.
- Li, Y., Chen, C., Zheng, X., Liu, J., and Wang, J. (2024). Making recommender systems forget: Learning and unlearning for erasable recommendation. *Knowledge-Based Systems*, 283:11124.
- Lin, F. and Reiter, R. (1994). Forget it ! In *Working Notes of AAAI Fall Symposium on Relevance*, pages 154–159.
- Liu, W., Wan, J., Wang, X., Zhang, W., Zhang, D., and Li, H. (2022). Forgetting fast in recommender systems.
- Lu, J., Wu, D., Mao, M., Wang, W., and Zhang, G. (2015). Recommender system application developments: A survey. *Decision Support Systems*, 74:12–32.
- Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., and Ren, X. (2019). Jointly learning explainable rules for recommendation with knowledge graph. In *The World Wide Web Conference*, pages 1210–1221. ACM.
- Matuszyk, P., Vinagre, J., Spiliopoulou, M., Jorge, A. M., and Gama, J. (2015). Forgetting methods for incremental matrix factorization in recommender systems. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC 2015. ACM.
- Matuszyk, P., Vinagre, J., Spiliopoulou, M., Jorge, A. M., and Gama, J. (2017). Forgetting techniques for stream-based matrix factorization in recommender systems. *Knowledge and Information Systems*, 55(2):275–304.
- Roy, D. and Dutta, M. (2022). A systematic review and research perspective on recommender systems. *Journal of Big Data*, 9(1).
- Schoenberg, I. J. (1969). *Publications of Edmund Landau*, page 335–355. Springer US.
- Tarus, J. K., Niu, Z., and Yousif, A. (2017). A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining. *Future Generation Computer Systems*, 72:37–48.
- Tavakolian, R., Taghi Hamidi Beheshti, M., and Moghadam Charkari, N. (2012). An improved recommender system based on forgetting mechanism for user interest-drifting. *International Journal of Information and Communication Technology Research*, 4(4):69–77.
- van Ditmarsch, H., Herzig, A., Lang, J., and Marquis, P. (2008). *Introspective Forgetting*, page 18–29. Springer Berlin Heidelberg.
- Verachtert, R., Michiels, L., and Goethals, B. (2022). Are we forgetting something? correctly evaluate a recommender system with an optimal training window. In *Proceedings of the Perspectives on the Evaluation of Recommender Systems Workshop*, volume 3228.
- Wang, Y., Wang, L., Li, Y., He, D., Liu, T.-Y., and Chen, W. (2013). A theoretical analysis of ndcg type ranking measures.
- Zhang, J. and Lu, X. (2020). A multi-trans matrix factorization model with improved time weight in temporal recommender systems. *IEEE Access*, 8:2408–2416.