

An overview of knowledge graph-based recommendation systems

Yutao Ye

Class 6, Senior Three, Beihu Campus, Chengdu Shishi Middle School, Chendu, China

2519684863@qq.com

Abstract. Recommendation systems have emerged as effective tools for mitigating information overload. Traditionally, recommendation systems employ various models such as Collaborative Filtering, Matrix Decomposition, and Logic Decomposition. Among these, Collaborative Filtering stands out due to its high efficiency. However, it encounters challenges related to cold start and sparse data. To address these challenges, the integration of Knowledge Graphs with recommendation systems has demonstrated significant advantages. This paper classifies Knowledge Graph-based recommendation systems into two categories: enhanced classical recommendation models and novel recommendation models integrated with Knowledge Graphs. We provide explanations for each category and compare them with traditional methods to draw conclusions. To inspire future research endeavors, this article identifies potential research areas and highlights unresolved issues.

Keywords: knowledge graph, recommendation system, graph neural network.

1. Introduction

In recent years, the rapid evolution of internet services, including e-commerce and streaming media, has underscored the pivotal role of recommendation systems in enhancing user experiences within these domains. Consequently, the optimization of recommendation system performance has emerged as a paramount challenge for enterprises operating in this arena. Extensive research efforts, spanning both academia and industry, have yielded a plethora of effective recommendation methods. The majority of these methods hinge on the bedrock of collaborative filtering, a technique that sieves information by analyzing user-item interactions and user feedback, aiming to unearth content that aligns with users' interests. Central to achieving robust recommendation performance is the availability of user-item interaction data, often considered the primary resource. However, this interaction data typically exhibits sparsity, given that users interact with only a fraction of the available items. This phenomenon, aptly labeled as the "sparsity problem," underscores the challenges encountered in developing collaborative filtering systems, the cornerstone of recommendation technology.

In light of technological advancements, knowledge graphs have emerged as a promising solution to combat the sparsity issue and other challenges faced by collaborative filtering systems. Knowledge graphs, renowned for their rich contextual relationships, offer a wellspring of information resources for recommendation systems. They play a pivotal role in alleviating data sparsity and harnessing complex, large-scale data effectively. Consequently, many researchers have integrated knowledge graphs into the recommendation process, leveraging them to uncover latent connections between users and items, thus

augmenting the wealth of observable data available for recommendations. Additionally, innovative models grounded in knowledge graph structures have been proposed to elevate the performance of recommendation systems.

This paper aims to provide a comprehensive survey of knowledge graph applications in recommender systems while comparing them to conventional methodologies, encompassing knowledge graph-based collaborative filtering systems and knowledge graph-based recommendation models. Furthermore, we delve into the trade-offs between knowledge graph embedding-driven recommendations and traditional recommendation approaches, offering constructive insights and highlighting avenues for future research.

The remainder of this paper is structured as follows: Section II furnishes a foundational overview of the topics and issues associated with recommender systems. Section III, IV, and V sequentially delve into traditional recommendation systems, knowledge graph-enhanced collaborative filtering systems, and recommendation models underpinned by knowledge graphs. Building on the analysis of empirical findings, Section VI proffers constructive trade-off considerations and identifies unresolved research questions. Finally, Section VII offers insights into graph embedding-based recommendations, encompassing contemporary challenges and potential remedies.

Section I introduces recommender systems and delineates their three primary approaches. Additionally, Section 2A provides a comprehensive outline of recommender systems, elucidating the problems they tackle. Subsequently, Section 2B expounds upon the concept of knowledge graphs and elucidates the rationale behind incorporating knowledge graph embeddings into recommender systems, setting the stage for the ensuing sections, namely Sections 3, 4, and 5.

2. Preliminaries

2.1. Recommender systems

The realm of recommender systems constitutes a vital subfield within computer software engineering. Leveraging cutting-edge technologies such as machine learning, these systems proactively present users with items aligning with their preferences while they navigate and engage with products, thereby fostering item consumption. The overarching objectives include conserving user time, elevating user experiences, optimizing resource allocation, and ultimately generating business value for service providers.

2.1.1. Definition

a) The recommendation system represents a software engineering solution that materializes recommendation capabilities through code, orchestrating the automated process of suggesting items to users.

b) The recommendation system exemplifies an application of machine learning. By assimilating user behavior data, formulating mathematical models, anticipating user interests, and ultimately proposing items that align with user preferences, it gratifies users' latent needs and enhances their overall experience.

c) The recommendation system manifests as an interactive product feature, seamlessly integrated into the product's framework. Users employ the product to initiate and activate the recommendation system. This system, in turn, furnishes users with personalized suggestions tailored to fulfill their latent needs. As a product component, the design of visual representation, user interaction mechanics, and the resolution of various contextual issues during the user-product interaction process require careful consideration.

d) The recommendation system embodies a (software) service. By leveraging the recommendation system, users access personalized item recommendations tailored to their unique interests, thereby satisfying individualized and passive user needs. Just as any service necessitates management and upkeep, operating a recommendation service is no exception. Addressing operational challenges and promoting service adoption often necessitates the involvement of human resources.

e) The recommendation system serves as a means to tackle information filtering and resource matching conundrums. It accomplishes this task through the application of machine learning algorithms and software engineering techniques, sieving and filtering information to cater to users in the vast landscape of information scenarios.

In light of the above exposition, we propose a refined definition: The recommendation system stands as a software service underpinned by machine learning principles. It deploys personalized recommendations for users by analyzing the intricate interplay between products and users, harnessing machine learning algorithms and software engineering practices to bolster recommendation efficiency and amplify user experiences.

2.1.2. Problem resolution

The advent of the internet, particularly the mobile internet, has precipitated the emergence of the recommendation system as a pivotal technological tool. Fundamentally, it serves as a rapid means of extracting information of interest from a vast sea of data when user preferences are unclear. The recommendation system amalgamates user-centric information (such as location, age, gender) with item-related data (including price, origin), and the user's historical interactions with items (purchase history, clicks, plays, etc.). Employing machine learning techniques, it constructs models of user interests, while employing software engineering methods to actualize software services, ultimately delivering precise personalized recommendations to users.

The recommendation system concurrently addresses the needs of item providers, platforms, and users. Consider Taobao, an e-commerce platform, where the item providers comprise thousands of individual shopkeepers, the platform is represented by Taobao itself, and users encompass individuals and entities engaged in shopping. Through the recommendation system, products can be more effectively showcased to users in need, thereby augmenting the efficiency of resource allocation within the social ecosystem.

In essence, the recommendation system resolves the resource allocation quandary. Through the amalgamation of software, algorithms, and engineering techniques, it harmonizes the supply side (item providers) with the demand side (users) via the intermediary platform (a mobile app offering personalized recommendations), with the overarching goal of enhancing resource allocation efficiency.

2.2. Knowledge graph

However, within various recommendation scenarios, especially in the context of big data, traditional recommendation methods are often constrained by the need for repetitive analysis of graph representations. This inherent limitation causes traditional recommendation methods to lag behind their knowledge graph-based counterparts in terms of scalability and adaptability. Embedded recommendation approaches, due to their reliance on rigidly pre-learned graph representations, struggle to effectively adapt to diverse recommendation scenarios, thereby encountering challenges associated with data sparsity.

In contrast, recommendation systems grounded in knowledge graph embeddings circumvent these limitations by directly leveraging node embedding vectors that encapsulate user and item characteristics, previously gleaned from graph representations. Furthermore, when coupled with machine learning techniques, knowledge graph embedding-based recommendation systems possess the capacity for pattern discovery, ultimately enhancing recommendation accuracy. This section aims to elucidate the fundamental concepts underpinning knowledge graphs and their applications in recommender systems.

2.2.1. Definition. A Knowledge Graph, also referred to as a Knowledge Domain Visualization or Knowledge Domain Mapping in the library and information domain, constitutes a series of graphical representations that delineate the evolution of knowledge and its structural relationships. Leveraging visualization technology, Knowledge Graphs serve as a means to depict knowledge resources, their carriers, as well as to mine, analyze, construct, map, and display knowledge and the intricate interconnections that underlie it.

2.2.2. Application in recommender systems. a) Knowledge Graph-Enhanced Classical Recommendation Models

b) Novel Recommendation Models Leveraging Knowledge Graphs

3. Traditional recommendation system

3.1. Collaborative filtering system

3.1.1. *Essence:* The Collaborative Filtering system comprises two fundamental algorithms:

a) User-based Collaborative Filtering Algorithm (UserCF): Recommends items to a user based on the preferences of other users with similar interests.

b) Item-based Collaborative Filtering Algorithm (ItemCF): Recommends items similar to those a user has previously liked.

3.1.2. *Introduction:* Among the various recommendation strategies, the collaborative filtering (CF) approach has achieved significant success by harnessing historical interactions or user preferences, enabling the effective filtering of vast datasets [1]. According to Mehdi Elahi et al., collaborative filtering recommender systems express user preferences through ratings, and typically, a higher number of user ratings leads to more reliable recommendations [2]. The collaborative filtering system filters information by scrutinizing the relationship between users and their feedback evaluations of items, with the goal of identifying content that aligns with a target user's interests. Users' preferences are represented as row vectors, item attributes as column vectors, and similarity is calculated through a scoring matrix that encompasses rows and columns. Various methods exist for calculating similarity, including:

- Cosine similarity
- Pearson correlation coefficient
- Euclidean distance
- Taxicab geometry

3.2. Overall Process:

(a) UserCF: This process involves quantifying user preferences for products, scoring previous products based on previous user actions such as purchases, collections, and additions to shopping carts. Similar users are then identified, and their scores are used to predict a user's preferences. Recommendations are made for products with higher predicted scores, while those with lower scores are not recommended. Lastly, the evaluation weight of user preferences for the new product is calculated, often employing Pearson correlation to standardize ratings among different users.

(b) ItemCF: ItemCF computes item similarities using historical preference data from all users. If many users exhibit simultaneous preferences for items A and B, it suggests a high degree of similarity between the two items. The Pearson correlation coefficient is often used for optimizing this process.

3.3. Technology Used: Similarity Calculation

Two primary methods for similarity calculation are as follows:

(a) Cosine Similarity:

This method measures the angle between two vectors, with smaller angles indicating greater similarity.

$$\text{sim}(i, j) = \cos(i, j) = \frac{i \cdot j}{\|i\| \cdot \|j\|} \quad (1)$$

$$\cos(\theta) = \frac{\sum_{k=1}^n x_{1k} x_{2k}}{\sqrt{\sum_{k=1}^n x_{1k}^2} \sqrt{\sum_{k=1}^n x_{2k}^2}} \quad (2)$$

(b) Pearson Correlation Coefficient:

This method is utilized when users tend to assign scores arbitrarily (e.g., one user consistently rates items highly, while another rates items poorly). Pearson correlation involves subtracting the mean values from both vectors before calculating cosine similarity, often yielding more accurate results.

$$\text{sim}(i, j) = \frac{\sum_{p \in P} (R_{i,p} - \bar{R}_i)(R_{j,p} - \bar{R}_j)}{\sqrt{\sum_{p \in P} (R_{i,p} - \bar{R}_i)^2} \sqrt{\sum_{p \in P} (R_{j,p} - \bar{R}_j)^2}} \quad (3)$$

3.4. Content-Based Filtering Algorithms

1) Introduction: Content-based filtering algorithms rely on the analysis of a user's historical behavior to model their interests, allowing for proactive recommendations that align with their preferences and needs.

2) Essence: These algorithms engage in direct analysis and calculation based on the inherent characteristics of both items and users. For instance, consider movie A as a sci-fi film; content-based filtering can recommend movie A to user B by scrutinizing the tags associated with sci-fi movies within user B's viewing history

3) The Overall Process (Using Movie Watching as an Example):

(a) Tagging the Movie: This involves extracting keywords from movie reviews or determining multiple tags for a movie based on commonly used tags. Subsequently, the TF-IDF (Term Frequency - Inverse Document Frequency) value is calculated for each label to ascertain label weight, with the top-K labels selected as the movie's tags.

(b) Establishing an Inverted Index: An inverted index is created to facilitate the retrieval of corresponding movies based on keywords.

(c) Tagging the User: By evaluating a user's historical viewing records, keywords associated with all previously watched movies are identified, word frequencies are tabulated, and the keyword with the highest frequency is designated as the user's interest word.

(d) Real-time Recommendations: Using the user's interest words, movies are retrieved from the inverted index to generate real-time recommendations.

Technology Used: TF-IDF Feature Extraction Technology (4):

$$w_{ij} = TF_{ij} \cdot IDF_i = \frac{f_{ij}}{f_{dj}} \cdot \log\left(\frac{N}{n_i}\right) \quad (4)$$

3.5. Matrix Factorization Model

3.5.1. Introduction. Matrix Factorization represents a direction stemming from collaborative filtering, devised to address the generalization challenges associated with processing sparse matrices. Notably, the Latent Factor Model, a subset within this category, relies on the co-occurrence matrix derived from collaborative filtering. This model employs denser hidden vectors (analogous to embeddings in NLP) to represent users and items. It delves into uncovering latent interests and features of users and items, ultimately addressing the generalization issue. The matrix is decomposed into the product of two low-rank matrices, with missing data being imputed via the inner product of these decomposed matrices. However, it is worth noting that incremental training can be challenging, particularly when faced with a surge in sample data, which may necessitate the rebuilding of the entire matrix.

Matrix Factorization involves breaking down the rating matrix of collaborative filtering into a user matrix multiplied by an item matrix, both represented as hidden vectors. When predicting the rating of a new item, one simply multiplies the item vector by the user vector. Various algorithms have emerged to perform this matrix decomposition, including BasicSVD, RSVD, ASVD, and SVD++.

4. Knowledge graph enhanced classical recommendation model

4.1. Translating models embedded in knowledge graphs

a. TransE (Translating Embeddings for Modeling Multi-relational Data)

TransE(5) stands as the inaugural model in the series of transformation models, grounded in the fundamental concept of minimizing the distance between the head vector and the relation vector to approximate the tail vector. It is crucial to note, however, that this model is primarily designed to handle one-to-one relationships and is less suited for addressing one-to-many or many-to-one relationships.

TransE

$$h + r \approx t \quad f_r(h, t) = \|h + r - t\| \quad (5)$$

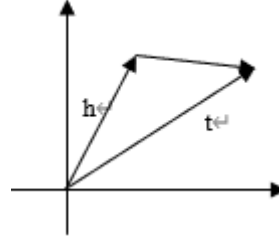
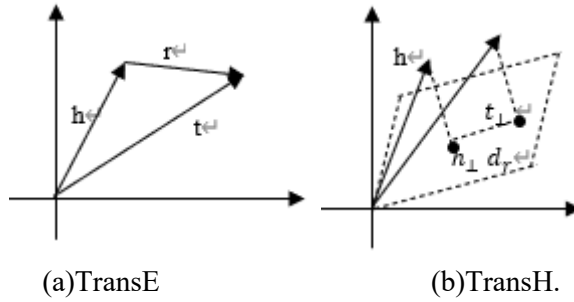


Figure 1. TransE.

b. TransH (Knowledge Graph Embedding by Translating on Hyperplanes)

To overcome the limitations of TransE when confronted with intricate relationships, such as one-to-many and many-to-many, Cao et al. introduced TransH(6), a knowledge graph embedding module, and a translation-based user preference model. This approach leverages the knowledge graph to enrich user preferences and representation projection [3], enabling the mining of more granular user preferences through joint learning, thereby enhancing recommendation efficacy. One of its key advantages lies in its ability to handle one-to-many, many-to-one, and many-to-many relationships without introducing added complexity or training challenges to the model.



(a)TransE

(b)TransH.

c. TransR (Learning Entity and Relation Embeddings for Knowledge Graph Completion)

TransE and TransH models operate on the assumption that entities and relations are vectors within a shared semantic space, leading to the implication that similar entities will closely cluster within this shared entity space. However, each entity may encompass various facets, and different relationships can emphasize distinct aspects of an entity. To tackle this challenge, the TransR model introduces a novel approach by modeling entities and relations within two separate spaces: the entity space and multiple relation spaces (relation-specific entity spaces). Transformations are then executed within the corresponding relation space, giving rise to the name TransR.

In various recommendation system applications, collaborative filtering systems often grapple with the issue of data sparsity. In response to this challenge, Zhang et al. recently proposed a hybrid recommendation framework. This framework capitalizes on heterogeneous information within the knowledge graph to enhance collaborative filtering quality [4]. The framework encompasses three types of knowledge stored within the knowledge graph: structural knowledge (comprising triple facts), textual knowledge (including textual summaries of books or movies), and visual knowledge (such as book covers or movie poster images). These sources of information collectively generate semantic representations of items. Structural knowledge is modeled using a conventional KG embedding

technique, TransR, to acquire structural representations for each item. Textual and visual knowledge, on the other hand, is extracted using stack denoising autoencoders and stack convolutional autoencoders, respectively.

$$\begin{aligned} h_{\perp} + d_r &\approx t_{\perp} \\ h_{\perp} &= h - w_r^T h w_r, \quad t_{\perp} = t - w_r^T h w_r \\ f_r(h, t) &= \|h_{\perp} + d_r - t_{\perp}\|_2^2 \end{aligned} \quad (6)$$

Translating entities to a hyperplane by a relation transformation matrix

4.2. Others

Top-N Collaborative Leaching Recommendation Algorithm

To address the issue of low recommendation accuracy, Ming Zhu et al. introduced a Top-N Collaborative Leaching Recommendation Algorithm rooted in knowledge graph embedding [5]. This algorithm elevates the semantic effectiveness of Collaborative Filtering recommendations and addresses the limitation of Collaborative Filtering algorithms, which typically overlook the intrinsic knowledge and information associated with the items themselves.

ECFKG (Explainable Collaborative Filtering over Knowledge Graph)

Ai et al. introduced the Explainable Collaborative Filtering over Knowledge Graph (ECFKG) based on knowledge graph [6]. ECFKG employs TransE to model various types of user behaviors and project attributes, encoding collaborative relationships into a graph structure. This approach extends the Collaborative Filtering algorithm and learns entity representations to uncover latent user preferences. Additionally, it features a soft matching algorithm designed to identify interpretation paths from users to projects. The best path is determined through a combination of breadth-first search and a soft matching formula, enabling the calculation of path probabilities to generate natural language explanations for recommendations.

Kopra (Knowledge Pruning-based Recurrent Graph Convolutional Network)

Tian et al. jointly developed Kopra, a knowledge pruning-based recurrent graph convolutional network, employing TransE to embed significant entities extracted from historical news headlines and abstracts [7]. The model employs recurrent graph convolution (RGC) to aggregate entity context information, constructing a user interest graph. RGC goes a step further by enhancing and adapting the user interest map. This is achieved by identifying highly correlated entities within the knowledge graph through a knowledge pruning strategy, resulting in the derivation of both long-term and short-term preference representations for users. This enhancement leads to more granular-level causal explanations for recommendations.

5. Novel recommendation model with knowledge graph

5.1. Entity 2 Rec

Enrico Palumbo et al. introduced "Entity 2 Rec," a model designed to learn user-project relevance by employing attribute-specific knowledge graph embedding [8]. A crucial aspect of using a knowledge graph for project recommendations lies in the ability to effectively define measures of user-item relevance within the graph.

5.2. Unsupervised API recommendation

Xin Wang et al. proposed an unsupervised API recommendation method founded on deep random walks across the knowledge graph [9]. Their approach involves constructing a comprehensive knowledge graph and subsequently expressing it in a specific manner. To enhance recommendation accuracy, they have devised an entity preference process to account for distinct entity preferences. API recommendations are generated through unsupervised feature learning.

5.3. KQGC (*knowledge-query-based graph convolution recommendation system*)

The combination of Knowledge Graph Embeddings (KGE) and Graph Neural Networks (GNNS), often referred to as KG-GNNS, has been explored in numerous academic works and has demonstrated effectiveness. Daisuke Kikuta et al. introduced a knowledge-query-based graph convolution recommendation system model [10]. In contrast to KG-GNNS, KQGC places a stronger emphasis on smoothing and employs a straightforward linear graph convolution to enhance KGE. Pre-trained KGE is directly integrated into KQGC, with the KGC being smoothed by aggregating knowledge queries from neighboring entities. This process aligns entity embeddings at relevant vector points, effectively enhancing the smoothness of KGE. The proposed KQGC is applied to recommend potential users for specific products, with its effectiveness validated through experimental results.

5.4. HAKG (*hierarchical attention knowledge graph embedding*)

Many existing methods require modifications to harness knowledge graphs accurately for obtaining user preferences. These methods either represent user-item connections through paths with limited expressive capabilities or implicitly model them by disseminating information across the entire knowledge graph, potentially resulting in erroneous data. In this study, Xiao Sha et al. introduced a novel Hierarchical Attention Knowledge Graph Embedding (HAKG) [11] framework designed for effective recommendation. HAKG initially extracts expressive subgraphs linking user-item pairs to capture their connectivity, aligning with the semantics and topology of the knowledge graph. These subgraphs are subsequently hierarchically encoded and attended to generate efficient subgraph embeddings, thus enhancing user preference prediction. Through extensive experiments, the researchers demonstrate the superiority of HAKG over existing recommendation methods and its potential to mitigate the data sparsity challenge.

6. Comparison and recommendation

6.1. *Graph-based recommender systems and traditional recommender systems*

There exists an ongoing debate concerning the effectiveness of knowledge graph embedding-based recommendations (discussed in Sections V and VI) in comparison to traditional recommender systems when it comes to effectively leveraging side information or knowledge to enhance recommendation efficiency. In terms of scalability [12,13], recommendations grounded in graph embeddings typically outperform traditional recommender systems. When confronted with vast and intricate datasets, knowledge graph-based recommendation systems harness the three characteristics of big data (Volume, Variety, and Velocity) with remarkable speed and efficiency to suggest items to users. This advantage stems from distinct theoretical foundations: After data (or information) organization is represented through a knowledge graph, traditional recommender systems rely on topological analysis characteristics. In contrast, recommender systems founded on knowledge graph embeddings operate through node embedding vectors, preserving features indicated by embedding techniques, thus eliminating the need for repetitive analysis, as seen in traditional recommendations [14]. Consequently, knowledge graph embedding-based recommendations demonstrate significantly enhanced scalability. Additionally, the storability of embedding vectors supports downstream machine learning tasks [15], requiring feature vectors of data instances as inputs, such as classification [16-18], link prediction [19], and more. This property of embedding vectors positions knowledge graph-based recommendations as superior to traditional recommendations in terms of model scalability.

However, in the context of model interpretability [20] (i.e., understanding why the model provides particular recommendations to users), knowledge graph embedding-based recommendations often lag behind traditional recommendations. This discrepancy arises because knowledge graph embedding-based recommendations predominantly employ machine learning methods [21], resulting in models that are mostly black boxes, relying heavily on input-output data to discover underlying patterns through numerical or analytical optimization techniques [22]. In contrast, traditional recommendations can offer direct interpretability. While recent studies have argued that interpretability of recommendation results

can be achieved indirectly through the utilization of content within knowledge graphs or causal learning (causal inference) [23] to reason and comprehend user preferences, the interpretability of recommendation models remains fundamentally constrained.

Moreover, the debate between graph embedding-based recommendations and traditional recommendations extends to recommendation accuracy. Undoubtedly, by incorporating auxiliary information and knowledge, recommendation methods founded on knowledge graph embeddings can notably enhance recommendation accuracy compared to traditional methods [24,25]. However, their ability to predict implicit user-item interactions in certain recommendation tasks may not match that of traditional recommendations, casting doubt on their overall performance. Similar findings have been reported by Dacrema et al. [26]. In reality, the development of graph embedding-based recommendations is not in opposition to traditional recommendations. Analyzing traditional recommendation models can inspire graph embedding-based recommendations to explore aspects such as motifs [27], subgraphs [28], and neighborhoods, ultimately improving their interpretability [29].

Conversely, knowledge graph-based recommendations have been applied in recent recommendation scenarios, such as Conversational Recommender Systems (CRS) or News Recommendation, offering promising application prospects for traditional recommendations as well. Therefore, a future-oriented approach would prioritize the simultaneous development of both aspects rather than a singular focus on one.

6.2. Knowledge graph-enhanced classical recommendation models and novel recommendation models with knowledge graph

Knowledge graph-enhanced classical recommendation models predict and recommend based on both user and project aspects. They construct a database of user-project preferences, linking it with the knowledge map to address the limitation of collaborative filtering recommendation algorithms, which often overlook the knowledge and information associated with the projects themselves [5].

Although the knowledge map enhances the efficiency of collaborative filtering systems, challenges such as cold start [30], data sparsity, and the synonym problem remain inevitable. These challenges have not been effectively resolved. In response, novel recommendation models with a knowledge graph have emerged to provide solutions to these issues. These models are built upon the foundation of the knowledge map and have significantly improved recommendation performance. They utilize feature learning from the knowledge graph to reduce the high dimensionality and heterogeneity of the knowledge graph, enhancing the flexibility of knowledge graph applications. This approach also reduces the workload associated with feature engineering and mitigates the potential computational burden introduced by knowledge graph integration. However, they face challenges related to being perceived as "black boxes" and having insufficient interpretability, as mentioned earlier. These challenges can only be effectively addressed by integrating novel recommendation models with knowledge graph-enhanced classical recommendation models.

Consequently, it is our belief that future development should encompass both aspects simultaneously, rather than exclusively focusing on one development trend.

7. Discussions and outlook

7.1. Current challenges

(1) Realizing Explainability in Graph Embedding-Based Recommendation: The challenge lies in making graph embedding-based recommendation systems more interpretable, ensuring that users can understand the reasoning behind recommendations.

(2) Deep Integration of Knowledge Graphs and Recommendation Systems: The quest for seamlessly combining knowledge graphs and recommendation systems is an ongoing challenge, requiring strategies to maximize their synergy.

(3) Effectiveness of Recommendation Categories in Different Contexts: Assessing whether all three recommendation categories (collaborative filtering, content-based, and hybrid) are universally effective in various contexts remains a complex issue.

(4) Enhancing Recommendation System Performance: The pursuit of improved recommendation systems necessitates exploring innovative methods and techniques beyond the current state of the art.

(5) Leveraging Graph Topology for Graph Embedding-Based Recommendation: Investigating how graph topology analysis can contribute to the efficacy of graph embedding-based recommendation systems is another area of interest.

7.2. Potential solutions

1) Interpretability: Efforts to enhance the interpretability of graph embedding-based recommendation systems are crucial for knowledge acquisition and real-world applications. Some progress has been made in this regard. For instance, TransF [31] incorporates a sparse attention mechanism to uncover hidden relational concepts and transfer statistical strength through concept sharing, leading to easily explainable associations between relations and concepts. CrossE [32], a novel knowledge graph embedding, explores knowledge graph explanation schemes by employing embedding-based path searching for generating link prediction explanations. While recent neural models have shown impressive performance, they often lack transparency and interpretability. Some methods address this limitation by blending black-box neural models with symbolic reasoning through the integration of logical rules, enhancing interpretability to instill trust in predictions. Further research should continue to enhance interpretability and boost the reliability of predicted knowledge.

2) Historical Preferences: Bei Hui et al. [33] introduce the KG-Aware recommendation model, which utilizes a self-attention mechanism to extract short-term and long-term user preferences from individual users' historical behaviors. These historical preferences are then deeply mined in combination with knowledge graphs, offering potential avenues to enhance recommendation systems.

References

- [1] B. Schafer, "LNCS 4321 - Collaborative Filtering Recommender Systems," 2007. [Online]. Available: <https://www.researchgate.net/publication/200121027>
- [2] M. Elahi, F. Ricci, and N. Rubens, "LNBIP 188 - Active Learning in Collaborative Filtering Recommender Systems." [Online]. Available: <http://www.unibz.it/http://www.uec.ac.jp>
- [3] Y. Cao, X. Wang, X. He, Z. Hu, and T. S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, Association for Computing Machinery, Inc, May 2019, pp. 151–161. doi: 10.1145/3308558.3313705.
- [4] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, Aug. 2016, pp. 353–362. doi: 10.1145/2939672.2939673.
- [5] M. Zhu, D. S. Zhen, R. Tao, Y. Q. Shi, X. Y. Feng, and Q. Wang, *Top-N collaborative filtering recommendation algorithm based on knowledge graph embedding*, vol. 1027. Springer International Publishing, 2019. doi: 10.1007/978-3-030-21451-7_11.
- [6] Q. Ai, V. Azizi, X. Chen, and Y. Zhang, "Learning heterogeneous knowledge base embeddings for explainable recommendation," *Algorithms*, vol. 11, no. 9, Sep. 2018, doi: 10.3390/a11090137.
- [7] Y. Tian et al., "Joint Knowledge Pruning and Recurrent Graph Convolution for News Recommendation," in *SIGIR 2021 - Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Association for Computing Machinery, Inc, Jul. 2021, pp. 51–60. doi: 10.1145/3404835.3462912.

- [8] E. Palumbo, D. Monti, G. Rizzo, R. Troncy, and E. Baralis, "entity2rec: Property-specific knowledge graph embeddings for item recommendation," *Expert Syst. Appl.*, vol. 151, pp. 1–48, 2020, doi: 10.1016/j.eswa.2020.113235.
- [9] X. Wang, X. Liu, J. Liu, X. Chen, and H. Wu, "A novel knowledge graph embedding based API recommendation method for Mashup development," *World Wide Web*, vol. 24, no. 3, pp. 869–894, 2021, doi: 10.1007/s11280-021-00894-3.
- [10] D. Kikuta *et al.*, "KQGC: Knowledge Graph Embedding with Smoothing Effects of Graph Convolutions for Recommendation".
- [11] X. Sha, Z. Sun, and J. Zhang, "Hierarchical attentive knowledge graph embedding for personalized recommendation," *Electron. Commer. Res. Appl.*, vol. 48, no. June, p. 101071, 2021, doi: 10.1016/j.elerap.2021.101071.
- [12] M. Singh, "Scalability and sparsity issues in recommender datasets: a survey," *Knowl. Inf. Syst.*, vol. 62, no. 1, pp. 1–43, 2020, doi: 10.1007/s10115-018-1254-2.
- [13] B. M. Sarwar, *Sparsity, scalability, and distribution in recommender systems*. University of Minnesota, 2001.
- [14] P. Goyal and E. Ferrara, "Graph Embedding Techniques, Applications, and Performance: A Survey," May 2017, doi: 10.1016/j.knosys.2018.03.022.
- [15] C. Yang, Z. Liu, C. Tu, C. Shi, and M. Sun, *Network embedding: theories, methods, and applications*. Springer Nature, 2022.
- [16] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node Classification in Social Networks," Jan. 2011, doi: 10.1007/978-1-4419-8462-3_5.
- [17] A. M. Khattak *et al.*, "Tweets classification and sentiment analysis for personalized tweets recommendation," *Complexity*, vol. 2020, 2020, doi: 10.1155/2020/8892552.
- [18] A. Galland and M. Lelarge, "Invariant embedding for graph classification," 2019. [Online]. Available: <https://hal.science/hal-02947290>
- [19] D. Liben-Nowell and J. Kleinberg, "The Link-Prediction Problem for Social Networks." [Online]. Available: www.arxiv.org.
- [20] Y. Zhang and X. Chen, "Explainable recommendation: A survey and new perspectives," *Foundations and Trends in Information Retrieval*, vol. 14, no. 1. Now Publishers Inc, pp. 1–101, Mar. 11, 2020. doi: 10.1561/15000000066.
- [21] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science (80-.)*, vol. 349, no. 6245, pp. 253–255, Jul. 2015, doi: 10.1126/science.aac4520.
- [22] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A Survey of Optimization Methods from a Machine Learning Perspective," Jun. 2019, [Online]. Available: <http://arxiv.org/abs/1906.06821>
- [23] D. Liang, L. Charlin, and D. M. Blei, "Causal Inference for Recommendation." [Online]. Available: <http://arxiv.org>
- [24] Y. Mao, S. A. Mokhov, and S. P. Mudur, "Application of Knowledge Graphs to Provide Side Information for Improved Recommendation Accuracy," Jan. 2021, [Online]. Available: <http://arxiv.org/abs/2101.03054>
- [25] B. Yu, C. Zhou, C. Zhang, G. Wang, and Y. Fan, "A Privacy-Preserving Multi-Task Framework for Knowledge Graph Enhanced Recommendation," *IEEE Access*, vol. 8, pp. 115717–115727, 2020, doi: 10.1109/ACCESS.2020.3004250.
- [26] M. F. Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? A worrying analysis of recent neural recommendation approaches," in *RecSys 2019 - 13th ACM Conference on Recommender Systems*, Association for Computing Machinery, Inc, Sep. 2019, pp. 101–109. doi: 10.1145/3298689.3347058.
- [27] H. Peng, J. Li, Q. Gong, Y. Ning, S. Wang, and L. He, "Motif-Matching Based Subgraph-Level Attentional Convolutional Network for Graph Classification." [Online]. Available: www.aaai.org
- [28] J. Zhao, X. Wang, C. Shi, B. Hu, G. Song, and Y. Ye, "Heterogeneous Graph Structure Learning for Graph Neural Networks," 2021. [Online]. Available: www.aaai.org

- [29] H. Wang, Y. Deng, L. Lü, and G. Chen, “Hyperparameter-free and Explainable Whole Graph Embedding,” Aug. 2021, [Online]. Available: <http://arxiv.org/abs/2108.02113>
- [30] J. Gope and S. K. Jain, “A survey on solving cold start problem in recommender systems,” in *Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2017*, Institute of Electrical and Electronics Engineers Inc., Dec. 2017, pp. 133–138. doi: 10.1109/CCAA.2017.8229786.
- [31] Q. Xie, X. Ma, Z. Dai, and E. Hovy, “An Interpretable Knowledge Transfer Model for Knowledge Base Completion,” Apr. 2017, [Online]. Available: <http://arxiv.org/abs/1704.05908>
- [32] W. Zhang, B. Paudel, W. Zhang, A. Bernstein, and H. Chen, “Interaction embeddings for prediction and explanation in knowledge graphs,” in *WSDM 2019 - Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, Association for Computing Machinery, Inc, Jan. 2019, pp. 96–104. doi: 10.1145/3289600.3291014.
- [33] B. Hui, L. Zhang, X. Zhou, X. Wen, and Y. Nian, “Personalized recommendation system based on knowledge embedding and historical behavior,” *Appl. Intell.*, vol. 52, no. 1, pp. 954–966, 2022, doi: 10.1007/s10489-021-02363-w.