Information Processing and Management 63 (2026) 104393

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

Learning path recommendation based on forgetting factors and knowledge graph awareness

Yunxia Fan [a], Mingwen Tong [a,*], Duantengchuan Li [b,**]

[a] Faculty of Artificial Intelligence in Education, Central China Normal University, Wuhan 430079, China
[b] School of Information Management, Wuhan University, Wuhan 430072, China

ARTICLE INFO

Keywords:
Learning path recommendation
Forgetting factor
Knowledge graph
Reinforcement learning

ABSTRACT

Learning path recommendation involves generating sequences of learning objects that are adapted to learners' needs, goals, abilities, and other factors through recommendation algorithms. Reinforcement learning (RL) has become an important approach for this task; however, it primarily emphasizes recommending new knowledge concepts while neglecting the necessity of revisiting forgotten ones. To overcome this limitation, FKGRec is introduced as a learning path recommendation framework that incorporates forgetting factors and knowledge graph awareness. To address the forgetting problem, a novel method named MemGNN is proposed, which integrates forgetting and knowledge graph features and employs a graph neural network with a memory gate structure to predict both new and previously learned knowledge concepts at each learning step. To further optimize the sequencing of new and previously learned knowledge concepts, an action space is constructed based on knowledge concept prediction, taking learners' cognitive states into account. An RL algorithm is then applied to recommend optimal learning paths by balancing new and previously learned knowledge concepts using a designed reward function. Experiments conducted on three datasets demonstrate that FKGRec surpasses existing state-of-the-art frameworks. A case analysis shows that the FKGRec framework can recommend learning paths that integrate new and previously learned knowledge concepts, aligned with learners' current cognitive state and forgetting factors.

1. Introduction

1.1. Motivation

In recent years, recommendation systems have become prevalent across a wide range of domains, including social media platforms (such as LinkedIn, Facebook, and Twitter), e-commerce sites (such as Taobao and Amazon), and online video platforms (such as Twitch, YouTube, and Netflix). These systems generally rely on users' historical behavior data to provide personalized recommendation services.

Matrix factorization is a commonly used collaborative filtering algorithm in recommendation systems. It estimates similarities between users and items, and the algorithm is regarded as efficient and simple (Luo, Zhou, Li, & Shang, 2018; Ma, Huang, Tang, & Zhang, 2022; Mu & Yuan, 2024). Nevertheless, matrix factorization typically depends on static assumptions and has difficulty adapting to dynamic variations in user interests. To address this issue, deep learning (DL)-based recommendation methods have

* Correspondence to: Central China Normal University, NO.152 Luoyu Street, Wuhan 430079, People's Republic of China.
** Corresponding author.
E-mail addresses: tmw@ccnu.edu.cn (M. Tong), dtclee1222@whu.edu.cn (D. Li).

https://doi.org/10.1016/j.ipm.2025.104393
Received 2 March 2025; Received in revised form 1 September 2025; Accepted 1 September 2025
Available online 19 September 2025

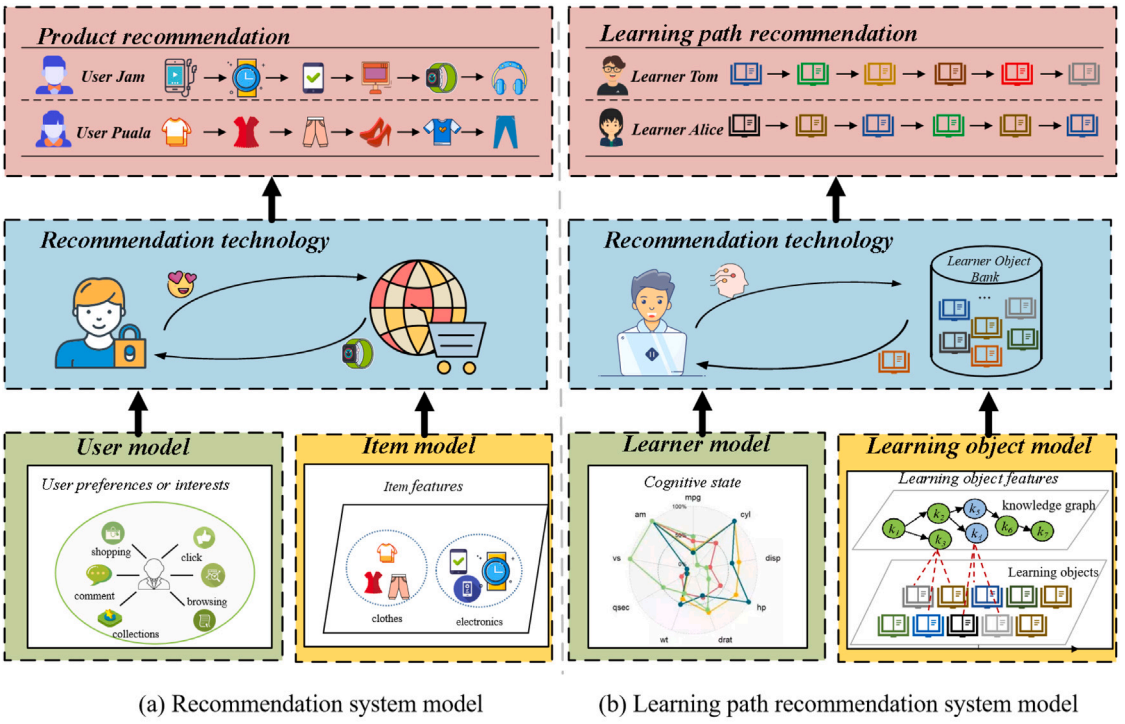


Fig. 1. Schematic overview of the recommendation system model.

been developed (Li et al., 2025; Liu et al., 2022). With strong feature learning capabilities, these methods are able to capture more complex patterns of user–item interactions and integrate different types of auxiliary information to enhance recommendation performance (Cheng, Khalitov, Yu, Zhang, & Yang, 2023; Duan, Zhang, Qiu, & Huang, 2022; Guo & Zeng, 2023; Huang & Wang, 2021; Sun et al., 2019; Wang, Xu, Yu, & Xu, 2020; Yuan, Tang, Yan, Hu, & Du, 2022). Although they have achieved notable success in modeling user interests, most DL approaches assume that such interests remain consistent over time, which limits their ability to reflect dynamic changes in user preferences. To overcome this challenge, DL and reinforcement learning (RL) are combined in recommendation systems. This integration enables dynamic adjustment of recommendation strategies through continuous interactions between users and items, thereby improving recommendation strategies through continuous interactions between users and items, thereby improving recommendation quality and user satisfaction (Sun, Zhuang, Zhu, He, & Xiong, 2021; Xu, Gao, Sheng, & Chen, 2021).

With the growth of online learning, an increasing number of researchers have attempted to apply recommendation algorithms in education to support learners through learning path recommendation (Ma et al., 2022, 2024; Mu & Yuan, 2024; Yun, Dai, An, Zhang, & Shang, 2024; Zhang, Hui et al., 2023; Zhang, Liu and Wang, 2023; Zhou, Huang, Hu, Zhu, & Tang, 2018). However, because learning path recommendation requires consideration of learners' cognition as well as the relationships between subject knowledge concepts, existing recommendation algorithms are often too complex to be directly applied for generating learning paths. Fig. 1 compares a general recommendation system with a learning path recommendation system. Fig. 1(a) illustrates the model of a general recommendation system, whereas Fig. 1(b) depicts the model of a learning path recommendation system. From this comparison, three key differences emerge: (1) **User modeling:** a general recommendation system primarily emphasizes user preferences and interest features, which remain relatively stable in the short term. In contrast, a learning path recommendation model focuses on the learners' cognitive state, which evolves continuously. (2) **Item modeling:** In a general recommendation system, items are typically classified, and there is no inherent sequence among them. In a learning path recommendation system, however, learning objects consist of multiple knowledge concepts, with prerequisite–successor relationships between them. Consequently, it is essential to model and constrain subject knowledge concepts effectively. (3) **Recommendation objectives:** In a general recommendation system, the objective is to satisfy users' current preferences and interests, while in a learning path recommendation system, the objective is to recommend learning objects that maximize learners' long-term outcomes. These objectives are fundamentally distinct.

The above analysis indicates that in the learning path recommendation task, learners' cognitive states are dynamic, learning objects are organized, and the recommended sequence must aim to maximize learners' long-term outcomes. Considering these characteristics, several studies have applied RL algorithms to learning path recommendation (Liu et al., 2019; Yun et al., 2024). In these approaches, the learners' cognitive state is treated as the "environment state", the knowledge graph is used to construct the "action space", and a reward function is designed. By training the strategy network, learning objects are recommended to the learner at each step. However, from a practical perspective, such algorithms only recommend learning objects associated with new

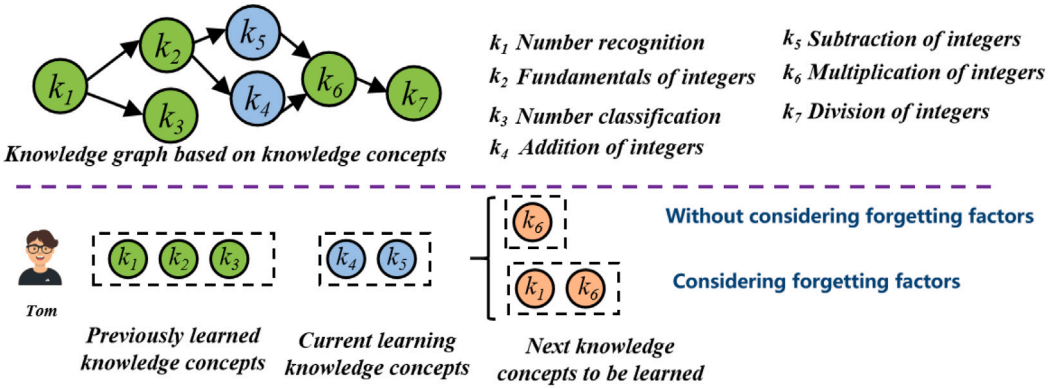


Fig. 2. Illustration of a recommended learning path.

knowledge concepts and are unable to include objects linked to forgotten concepts. Consequently, they do not address learners' requirements for both review and exploration. Furthermore, an analysis of RL-based learning path recommendation frameworks shows that this limitation arises because the action space is generally defined using successor knowledge concepts in the knowledge graph, which prevents the action space from reflecting forgotten concepts. As a result, it becomes difficult to recommend a reasonable learning path. For example, consider the learning path illustrated in Fig. 2. A learner, *Tom*, is studying a learning object containing both k_4 (addition of integers) and k_5 (subtraction of integers). During the learning process, the learner forgets k_1 (number recognition). If the forgetting factor is not incorporated and the action space is generated only from the knowledge graph at that point, it will include only the learning object associated with k_6 (multiplication of integers). In contrast, when both the forgetting factor and the knowledge graph are considered, the action space should include learning objects related to both k_1 (number recognition) and k_6 (multiplication of integers). Therefore, it is necessary to reconstruct the action space by integrating the learners' forgetting factor with the hierarchical relationships among knowledge concepts, enabling learners to revisit forgotten concepts at appropriate times and thereby ensuring more reasonable learning path recommendations.

Based on this, FKGRec is introduced as a learning path recommendation framework that incorporates forgetting factors and knowledge graph awareness. The framework comprises four main modules—a data preparation module, a cognitive diagnosis module, a knowledge concept prediction module, and a decision-making module. Among these, the knowledge concept prediction module and the decision-making module are the central components of FKGRec. In the knowledge concept prediction module, a novel method named MemGNN is proposed, which integrates forgetting features with knowledge graph features. This method predicts the knowledge concepts to be studied at each step of the learning process, ensuring that both new and previously learned concepts are included, thereby expanding the coverage of the action space. In the decision-making module, suitable learning objects—whether representing new or forgotten concepts—are selected from the current action space according to the learners' cognitive state. A reward function is then designed, and an RL algorithm is applied to recommend a learning path that maximizes long-term outcomes. This approach ensures that forgotten knowledge concepts are revisited at appropriate times and that both new and previously learned concepts are recommended in a reasonable order, thereby maintaining the scientific validity of the learning path.

1.2. Research objective and contributions

The main contributions of this study are summarized as follows:

- The FKGRec framework is proposed, which integrates knowledge concept prediction with RL algorithms to generate learning paths that address learners' requirements for both review and exploration.
- To overcome the limitation that the action space does not capture forgotten knowledge concepts, a novel knowledge concept prediction method is introduced based on the MemGNN model. This method combines forgetting features with knowledge graph features, predicts knowledge concepts at each step of the learning process, and incorporates forgotten concepts into the action space.
- By considering the learning order of new and previously learned knowledge concepts, an RL algorithm is applied to recommend learning objects that maximize long-term benefits according to the learner's cognitive state. This approach ensures that the recommended learning path follows a logical sequence for both new and previously learned knowledge concepts.
- The MemGNN method and the FKGRec framework are validated on three datasets, and the experimental results demonstrate that the MemGNN method achieves strong prediction performance. Meanwhile, the FKGRec framework surpasses all other state-of-the-art recommendation frameworks.

The remainder of this article is organized as follows. Section 2 reviews related work on recommendation technology and identifies the research problem. Section 3 defines key terminology and further clarifies the research problem. Section 4 introduces the FKGRec framework and describes its implementation in detail. Section 5 outlines the experimental design and presents the results. Section 6 discusses the experimental findings. Section 7 concludes the study.

2. Related work

Learning path recommendation involves generating an ordered sequence of learning objects through recommendation methods that account for learners' needs, goals, and abilities. Existing techniques for learning path recommendation have evolved through three main stages of development: approaches based on evolutionary algorithms, machine learning algorithms, and RL algorithms.

2.1. Learning path recommendation based on evolutionary algorithms

Evolutionary algorithms treat the learning path recommendation problem as a search optimization task. These methods embed learners' prior knowledge into the evolutionary computation model, apply predefined heuristic rules and optimization strategies, and simulate the generation of learning paths by drawing on natural evolutionary mechanisms. The algorithms determine the search direction and iteratively produce the optimal path. Representative approaches include genetic algorithms (Hwang, Kuo, Yin, & Chuang, 2010), ant colony optimization algorithms (Vanitha & Krishnan, 2019), and differential evolution algorithms (Wang, Zhang, Chen, Wang, & Xu, 2022). For example, Kurilovas, Zilinskiene, and Dagiene (2014) improved an ant colony optimization algorithm and applied it to adaptive learning path recommendation, enabling learners to achieve their learning goals according to individual learning preferences. Similarly, Benmesbah, Lamia, and Hafidi (2023) proposed an enhanced genetic algorithm that recommends adaptive learning paths tailored to learners' needs and contexts. This method reduced the search space and improved recommendation efficiency. Although evolutionary algorithms offer strong global search capabilities and can effectively identify optimal path combinations, their iterative processes are computationally expensive and unsuitable for real-time applications. Furthermore, because they rely on one-sided analysis and shallow modeling of learner needs, they are limited in accurately capturing personalized learning requirements, which restricts their effectiveness in practical recommendation systems.

2.2. Learning path recommendation based on machine learning

Machine learning algorithms address the learning path recommendation problem as a prediction task. By performing representation learning on large-scale educational data, they construct automated models capable of predicting appropriate learning paths. Machine learning approaches can be broadly divided into traditional methods and DL-based methods.

Traditional machine learning algorithms generally begin by clustering learners, followed by mining and analyzing implicit learning behavior patterns within different learner groups. Learning paths are then generated by identifying and referencing similar groups of learners. Commonly applied algorithms include association rule mining (Ma, Wang, Zhang, Liu, & Jiang, 2023), decision trees (Intayoad, Becker, & Temdee, 2017), and collaborative filtering (Mu & Yuan, 2024). For example, Ma et al. (2023) employed an association rule algorithm to extract sequences of knowledge concepts from learners' data and then applied a swarm intelligence algorithm to align each concept with suitable learning objects, thereby supporting the generation of adaptive learning paths. Similarly, Mu and Yuan (2024) developed a knowledge concept rating matrix based on learners' behavioral data, identified similar learner groups, and used the TransH model to embed the cognitive map into a vector space. An item-based collaborative filtering algorithm was subsequently applied to predict future scores of knowledge concepts, enabling personalized learning path recommendations. Traditional machine learning–based approaches are more objective and data-driven than evolutionary algorithms. These algorithms are characterized by simple models and fast training speeds. However, their data processing and model fitting are often rigid and lack flexibility, making them insufficient to meet the diverse learning needs of individual learners.

With the advancement of recommendation technologies, DL models have been increasingly adopted in learning path recommendation owing to their strong representation learning capabilities. These models can effectively capture complex patterns and nonlinear relationships within learner behavior sequences. Typically, DL approaches formulate the learning path recommendation task as a prediction problem, aiming to build automated models by applying representation learning to large-scale learner behavior data. In recent years, DL algorithms, such as long short-term memory (LSTM), Transformer and GNN models have been applied to learning path recommendation. For example, Zhou et al. (2018) first clustered learners and then trained an LSTM model to predict learners' complete learning paths and academic performance, subsequently selecting the most suitable path for each learner. Kang and McAuley (2018) proposed an attention-based sequential recommendation model that extracts relevant items from learners' interaction histories, assigns weights to them, and predicts the next item in the sequence. Gu (2025) designed two graph attention models: one calculated learners' preference for each feature of learning resources, while the other measured the correlation between user feature vectors and multiple learning resource sequences, thereby improving the accuracy of learning path recommendations. Zhang, Liu et al. (2023) constructed a multidimensional knowledge graph, enhanced the graph convolutional neural network, and evaluated the importance of learning resources through their features in the knowledge graph to better capture learners' preferences and recommend learning paths. DL-based learning path recommendation algorithms demonstrate strong expressive power and are well suited for training on large-scale datasets. However, these algorithms typically generate a complete learning path that reflects only the learner's initial cognitive state; they neglect the dynamic nature of cognitive states during the learning process. Changes arising from the new knowledge acquisition and forgetting alter cognitive states, yet static generation methods are unable to capture these changes or update learning paths in real time.

2.3. Learning path recommendation based on reinforcement learning

RL algorithms are particularly suitable for decision-making in dynamic environments (Lin, Li, Li, Chen, Liu et al., 2024; Lin, Li, Li, Chen and Wu, 2024). In the exploration phase, the agent interacts with the environment through an "observation-action-feedback" loop, recording actions that maximize cumulative rewards to form a policy network. In the decision-making phase, the agent can directly sense the environment and make optimal long-term decisions without further exploration. Given the variability of learners' cognitive states and the need to optimize long-term outcomes, RL algorithms have been applied to learning path recommendation (Liu, Wu, Chang and Gu, 2022; Xu et al., 2022).

In this context, the learning process is typically modeled as a Markov decision process (MDP), where the agent represents the recommender, the state corresponds to the learner's cognitive state, and the recommended learning objects serve as actions. A pedagogically meaningful reward function is then designed to enable the agent to continuously optimize the learning path recommendation strategy. For environmental state computation, studies use either single-type or multi-type behavioral data. For example, Luo, Gu, Dong, and Zhou (2025) employed only learners' historical behavior data and modeled the state with an LSTM network, whereas Ma et al. (2024) incorporated multi-source data, including knowledge backgrounds, learning preferences, and learning styles, and applied a multi-behavior Transformer with a knowledge graph for path recommendation.

Regarding action space definition, most research leverages knowledge graphs, where the successor knowledge concepts of the current concept constitute the action space (Liang, Mu, Chen, & Xie, 2022; Liu et al., 2019). To reduce the search space, Yun et al. (2024) constructed candidate sets of knowledge concepts based on predecessor-successor relationships in the knowledge graph, treating them as the action space at each step. The reward function is typically task-oriented, designed to reinforce desirable recommendation outcomes and guide policy learning.

Based on these studies, it is evident that knowledge graphs play a central role in organizing learning objects for path recommendation. Existing research generally defines the action space as the successor concepts of the current node, thereby constraining and guiding the recommendation process through graph structure. However, this approach remains limited, as it primarily recommends new concepts while overlooking the review of forgotten ones. Consequently, current RL-based systems often fail to meet learners' dual needs for review and exploration, largely due to action space design that neglects the forgetting factor.

To summarize, existing learning path recommendation methods increasingly integrate diverse features to enhance user modeling and better address learners' needs. During path generation, there is also greater emphasis on dynamically perceiving the learning environment, allowing adaptive adjustment of learning paths. Nevertheless, most approaches neglect the role of forgetting, producing recommendations that emphasize new knowledge while disregarding the review of previously learned but forgotten content. In contrast, the proposed FKGRec framework incorporates forgetting factors with knowledge graph features to generate learning paths that balance the acquisition of new concepts with the review of prior knowledge, based on learners' real-time cognitive states. The paths generated by FKGRec not only meet learners' review requirements but also maintain the logical progression of both new and previously acquired knowledge.

3. Preliminary

3.1. Definition of terminology

Cognitive state: The cognitive state denotes the learner's level of mastery over all knowledge concepts at a given time. Based on a learner's historical learning record, knowledge tracing algorithms can predict mastery of all concepts at the next time step (Gan, Sun, Peng, & Sun, 2020). Formally, the cognitive state is represented as $\mathbf{c} = [c_{k_1}, c_{k_2}, \ldots, c_{k_{|\mathcal{K}|}}]$, where $c_{k_i} \in [0, 1]$ indicates the probability that the ith knowledge concept is mastered.

Knowledge concept coverage: Knowledge concept coverage refers to the set of knowledge concepts associated with a learning object, denoted as $\mathcal{C}$. During the learning process, the coverage of a learner's next learning object can be predicted using DL algorithms (Ma et al., 2022; Ren, Liang, Shang, & Zhang, 2023).

Knowledge graph: A knowledge graph is essentially a directed acyclic graph in which nodes represent knowledge concepts and edges represent relationships between them. Formally, a knowledge graph is defined as $\mathrm{KG} = (\mathcal{K}, \mathcal{KR})$, where $\mathcal{K}$ is the set of knowledge concepts and $\mathcal{KR}$ is the set of relationships.

Learning goal: The learning goal is the knowledge concept a learner is expected to master in the course of learning, denoted as $target$. The learning process is considered complete once the learner masters this concept.

Forgetting factor: Forgetting refers to the inability to recall or recognize previously learned material, or the occurrence of errors in recall or recognition, which can hinder learning efficiency. Since forgetting is an unobservable variable, it is typically inferred indirectly through observable indicators, referred to as forgetting factors in this study. Following the approach in Nagatani et al. (2019), forgetting is modeled as $\lambda = [\Delta t, \Delta s, \Delta r]$, where Δt represents the time interval for revisiting the same concept, Δs the time interval across concept sequences, and Δr the number of times the same concept has been learned.

3.2. Problem formulation

The learning environment comprises multiple learners and learning objects. The learner set, denoted as $\mathcal{S} = \{s_1, s_2, \ldots, s_i\}$, represents a group of learners. The learning object bank (LOB), denoted as $\mathcal{LOB} = \{lo_1, lo_2, \ldots, lo_j\}$, is a collection of learning

Table 1
Key notations used in this study.

Notations	Description
$\mathcal{S} = \{s_1, s_2, \ldots, s_i\}$	Set of learner(s); m is the number of learners, s_i is the learner number
$\mathcal{LOB} = \{lo_1, lo_2, \ldots, lo_j\}$	Set of learning object(s); n is the number of learning objects
$\mathcal{K} = \{k_1, k_2, \ldots, k_{\|K\|}\}$	Set of knowledge concept(s); $\|K\|$ is the number of concepts
$\mathbf{lo}_j = [lo_j^{k_1}, lo_j^{k_2}, \ldots, lo_j^{k_{\|K\|}}]$	Vector representation of the jth learning object K_i is the ith concept
lo_j	Learning object number
dif_{lo_j}	Difficulty of learning objects lo_j
$\mathcal{X} = \{x_1, x_2, \ldots, x_{\|v\|}\}$	Complete learning records; $\|v\|$ is the length of the record, $x_{\|v\|}$ consists of s_i, lo_j, and b_{ij}, the value of b_{ij} is 0 or 1
$\mathbf{c} = [c_{k_1}, c_{k_2}, \ldots, c_{k_{\|K\|}}]$	Knowledge mastery vectors of the learner
$target$	Learning goal
$\mathbf{p} = [p_{k_1}, p_{k_2}, \ldots, p_{k_{\|K\|}}]$	Knowledge concept coverage vector
$\mathcal{C}$	Knowledge concept coverage set
$\mathbf{AB}$	Adjacency matrix representation of the knowledge graph
$\lambda = [\lambda_{k_1}, \lambda_{k_2}, \ldots, \lambda_{k_{\|K\|}}]$	Forgetting vectors for the knowledge concept k_i
$\mathcal{LP} = \{lo_1, lo_2, \ldots, lo_i\}$	Learning path
$\mathbf{state} = [s_i, lo_j, target, \mathbf{c}]$	Environmental state vector
$\mathcal{A}$	Action space
R	Aggregate reward
R_{step}	Reward for the agent in the exploration phase
R_{final}	Reward for the agent at the end of the exploration

objects. Each learning object involves one or more knowledge concepts, represented as nodes in the subject knowledge graph, with antecedent and successor relationships between concepts. Learners generate historical learning records, denoted as $\mathcal{X}_{s_i} = \{x_1, x_2, \ldots, x_{|v|}\}$. Each learner sets a learning goal before beginning the learning process.

The task is to recommend a learning path $\mathcal{LP} = \{lo_1, lo_2, \ldots, lo_i\}$ step by step, based on the learners' historical records $\mathcal{X}$ and learning goal $target$. The path should ensure that learners acquire new knowledge while reviewing previously learned content at appropriate times to mitigate forgetting. Ultimately, the objective of the learning path $\mathcal{LP}$ is to maximize the learners' overall achievement throughout the learning process.

A list of important notations used in this article is illustrated in Table 1, together with their descriptions, to aid readers in understanding the context.

4. Proposed FKGRec framework

In response to the above problem, the FKGRec framework has been developed, as shown in Fig. 3. The framework is composed of four main components: a data preparation module, a cognitive diagnosis module, a knowledge concept prediction module, and a decision-making module. The data preparation module provides the essential input for the framework. The cognitive diagnosis module, implemented with an LSTM model, dynamically tracks learners' cognitive state over time. The LSTM output is the probability that a learner has mastered each knowledge concept at moment t, denoted as $\mathbf{c}^t$. The knowledge concept prediction module, powered by the MemGNN method, forecasts the knowledge concepts to be learned at each step. Its output is the probability of the occurrence of knowledge concepts at moment t, denoted as $\mathbf{p}^t$. These two modules operate independently without direct interaction, and their outputs $\mathbf{c}^t$ and $\mathbf{p}^t$ jointly serves as inputs to the decision-making module. In this module, RL is employed to ensure a balanced sequence of new and previously learned knowledge in the recommended learning path. Specifically, $\mathbf{c}^t$ and **target** define the environment state $\mathbf{state}^t$, the action space $\mathcal{A}^t$ is generated based on $\mathbf{p}^t$, and corresponding rewards R^t are designed. Through a dynamic RL interaction mechanism, the final learning path is generated.

4.1. Data preparation

In the data preparation module, the source data required for the FKGRec framework are analyzed and structured. The primary input data consist of historical learning records, forgetting representation data, and knowledge graph data. The preparation process involves several steps. First, the core elements of the learning environment, learners, learning objects, and knowledge concepts, are extracted and represented. Specifically, a learner is denoted as s_i, a learning object as lo_j, and a knowledge concept as k_z. Since each learning object encompasses multiple knowledge concepts, the embedding of a learning object is represented as $\mathbf{lo}j = [lo_j^{k_1}, lo_j^{k_2}, \ldots, lo_j^{k|K|}]$. Subsequently, the learner's historical learning record, forgetting factors, and knowledge graph are modeled. The historical learning record is expressed as $\mathcal{X}s_i = x_1, x_2, \ldots, x|v|$. The modeling of forgetting factors and the knowledge graph is detailed in Section 3.1.

4.2. Cognitive diagnosis module

Within the cognitive diagnosis module, LSTM models are widely applied to predict learners' mastery of knowledge concepts and to monitor their cognitive states, and this approach has reached a relatively mature stage with high predictive accuracy (Ma et al.,

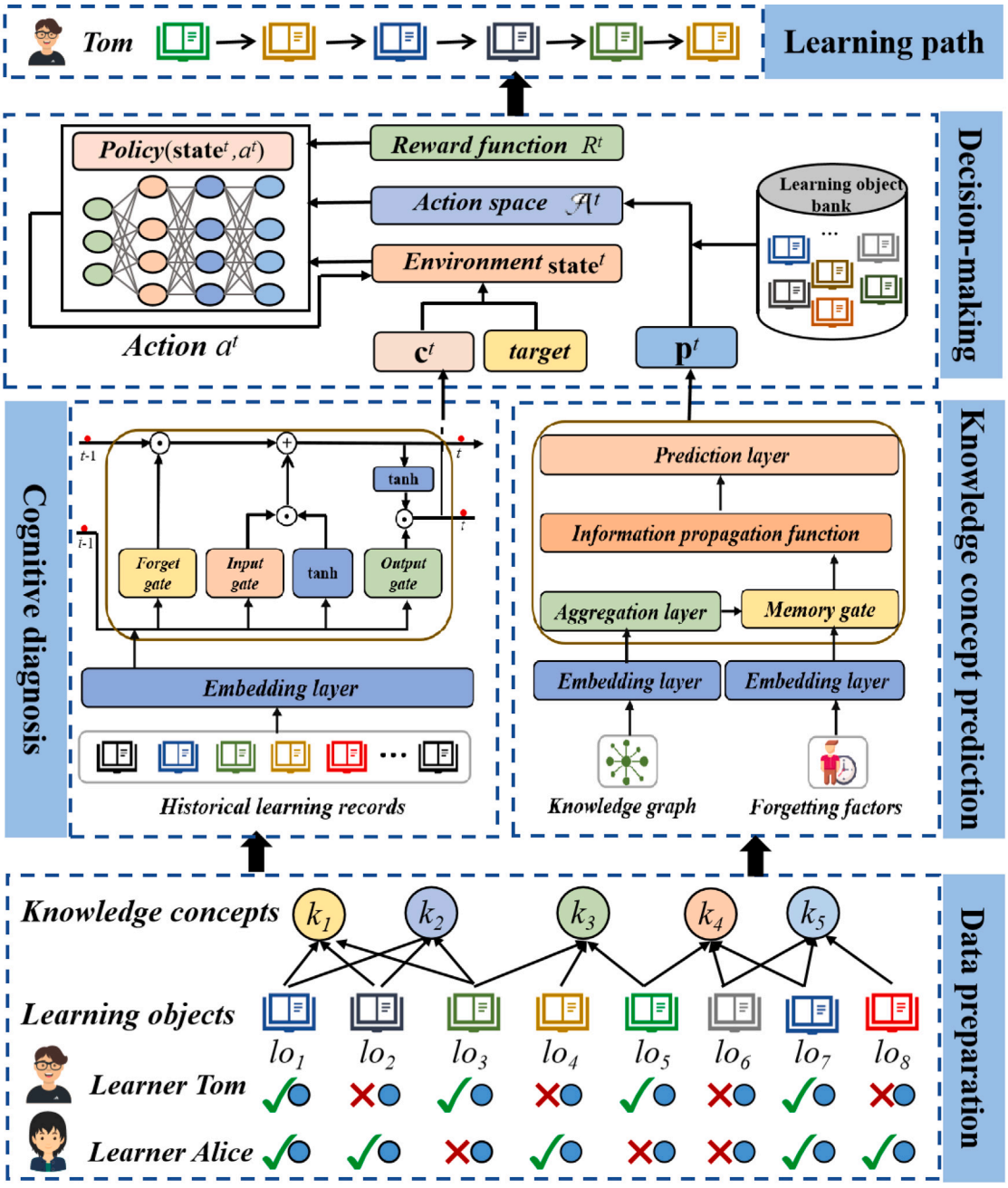


Fig. 3. The FKGRec framework comprises four modules: (1) the data preparation module, which analyzes and processes the source data; (2) the cognitive diagnosis module, which dynamically monitors learners' cognitive states; (3) the knowledge concept prediction module, which forecasts the knowledge concepts learners should study at each step; and (4) the decision module, which generates learning path recommendations. Collectively, these modules enable the generation of personalized learning paths.

2022; Yun et al., 2024). These models are further employed for dynamically tracking learners' mastery of knowledge concepts. The input to the LSTM model is the historical learning sequence $\mathcal{X}_{s_i} = \{x_1, x_2, \ldots, x_{|v|}\}$, where each element $x_{|v|} = (s_i, lo_j, b_{ij})$ denotes a learner's interaction with a learning object, and b_{ij} represents the learner's score for that object (0 or 1). During model training, the parameters are updated from Eqs. (1) to (6).

$$i^t = \sigma(\mathbf{W}_{xi}x^t + \mathbf{W}_{hi}h^{t-1} + \mathbf{b}_i), \tag{1}$$

$$f^t = \sigma(\mathbf{W}_{xf}x^t + \mathbf{W}_{hf}h^{t-1} + \mathbf{b}_f), \tag{2}$$

$$o^t = \sigma(\mathbf{W}_{xo}x^t + \mathbf{W}_{ho}h^{t-1} + \mathbf{b}_o), \tag{3}$$

$$cell^t = f_t cell^{t-1} + i^t \tanh(\mathbf{W}_{xc}x^t + \mathbf{W}_{hc}h^{t-1} + \mathbf{b}_c), \tag{4}$$

$$h^t = o^t \tanh(cell^t), \tag{5}$$

$$\mathbf{c}^t = \sigma(\mathbf{W}_{oL}o^t + \mathbf{b}_L). \tag{6}$$

At time step t, the **input gate** i^t (Eq. (1)) and the **forget gate** f^t (Eq. (2)), regulate the extent to which the previous cell state $cell^{t-1}$ is preserved and the amount of new information incorporated. The **output gate** o^t (Eq. (3)) controls which parts of the current cell state are transferred to the hidden state h^t. The **cell state** $cell^t$ (Eq. (4)) is updated by combining the retained memory $f^t cell^{t-1}$ with the candidate information $i^t \tanh(W_{xc}x^t + W_{hc}h^{t-1} + \mathbf{b}_c)$, thereby integrating past learning history with the present interaction. The updated cell state is then used to compute the hidden state h^t (Eq. (5)) through modulation by the output gate. Finally, the hidden state is passed to a fully connected layer with a sigmoid activation function (Eq. (6)) to generate the vector $\mathbf{c}^t$, where each element corresponds to the predicted mastery level of a knowledge concept at time t. This sequence allows the model to continuously monitor learners' knowledge states and forecast their subsequent performance.

4.3. Knowledge concept prediction module

In this study, the action space of the RL-based learning path recommendation framework is defined by the successor knowledge concepts of the current ones in the knowledge graph. However, this approach overlooks concepts that the learner may have forgotten. To address this issue, the FKGRec framework incorporates a knowledge concept prediction module that combines forgetting features with knowledge graph features. Within this module, the MemGNN method is introduced to predict the knowledge concepts learners should study at each step. This ensures that the prediction encompasses both new and forgotten concepts, thereby broadening the coverage of the action space. The overall architecture of MemGNN, shown in Fig. 4, comprises four key layers—embedding, aggregation, update, and prediction.

4.3.1. Embedding layer

The embedding layer extracts forgetting features and knowledge graph features from historical learning records and converts them into a unified low-dimensional embedding representation. The knowledge graph node features are represented by two matrices, W_x and W_c. Specifically, $W_x \in \mathbb{R}^{v \times d}$ encodes the knowledge concepts and score results of the historical learning objects, while $W_c \in \mathbb{R}^{2|v| \times d}$ captures the embeddings of knowledge concepts from the current learning object. Furthermore, the forgetting features and knowledge graph features are represented by the matrices W_f and **AB**, respectively.

4.3.2. Aggregation layer

The aggregation layer gathers the hidden states of the current knowledge concept together with those of its neighboring nodes. After aggregation, the embedding of each knowledge concept is updated to include more comprehensive graph structural information. For each knowledge concept k_i in the knowledge graph, two cases are distinguished during aggregation. If k_i belongs to the coverage set C^t of knowledge concepts for the current learning object lo_j, the embedding vectors $\mathbf{z}^t_{k_i}$ of k_i at time t is aggregated with the embedding vectors corresponding to C^t. Otherwise, the embedding vector of k_i is aggregated directly with its own representation. As illustrated in Fig. 2, the learning object lo_j at time t covers only knowledge concepts k_4 and k_5, therefore, $C^t = \{k_4, k_5\}$. The embedding vectors of k_4 and k_5 are denoted as $\mathbf{l}^t$, which serve as the input to the aggregation layer. The embedded hidden vector $\mathbf{z}^{t'}_{k_i}$ of knowledge concept k_i at time t is then computed as shown in Eq. (7):

$$\mathbf{z}^{t'}_{k_i} = \begin{cases} [\mathbf{z}^t_{k_i}, \mathbf{l}^t\mathbf{W}_x] & k_i \in C^t \\ [\mathbf{z}^t_{k_i}, \mathbf{W}_c(k_i)] & k_i \notin C^t \end{cases}, \tag{7}$$

where $\mathbf{W}_x$ and $\mathbf{W}_c$ are knowledge concept embedding matrices, and $\mathbf{W}_c(k_i)$ denotes the ith row of $\mathbf{W}_c$.

4.3.3. Update layer

The update layer involves two main processes: constructing the memory gate and defining the information propagation functions. The memory gate combines the forgetting vector $\lambda^t_{k_i}$ at time t with the embedded hidden vector $\mathbf{z}^{t'}_{k_i}$ of the knowledge concept λ_{k_i} to produce the embedded hidden vector $\mathbf{m}^{t'}_{k_i}$ incorporating forgetting features. The information propagation function then integrates the adjacency matrix **AB** with the forgetting vector λ_{k_i} to update the embedding vector $\mathbf{z}^{t+1}_{k_i}$ of the knowledge concept k_i for the next time step.

4.3.3.1. Construction of memory gate. Learners often forget knowledge during the learning process, and the memory gate in the neural network allows selective retention or forgetting of information, controlling information flow. For example, if a learner has

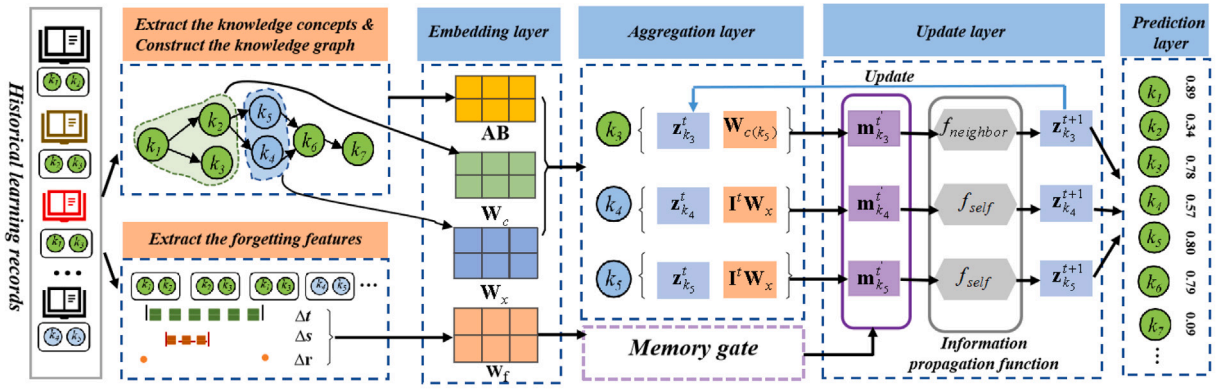


Fig. 4. The overall architecture of the MemGNN model comprises five components: (1) Feature extraction, which obtains knowledge graph and forgetting features; (2) Embedding layer, transforming extracted features into a unified low-dimensional embedding; (3) Aggregation layer, which aggregates embeddings of each knowledge concept to generate rich graph structure information; (4) Update layer, fusing forgetting features with knowledge graph features; and (5) Prediction layer, which forecasts the knowledge concepts learners should study at each step.

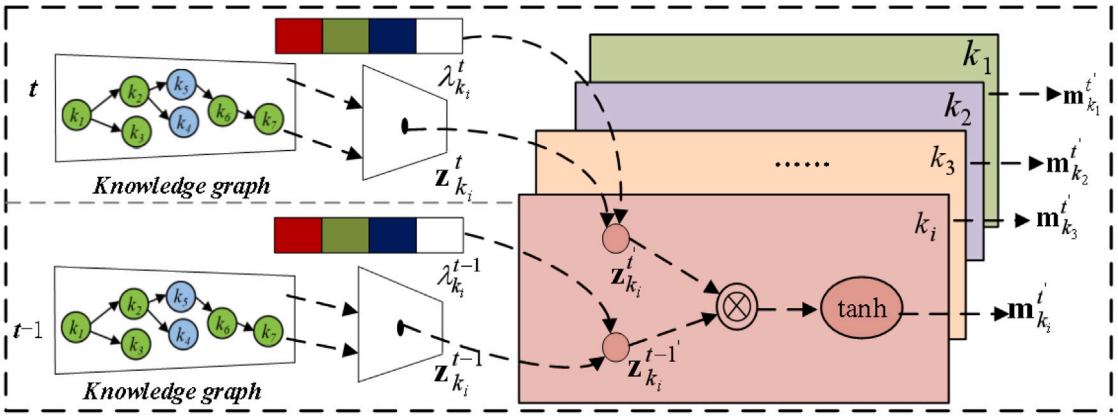


Fig. 5. Structure of the memory gate.

forgotten the knowledge concept k_1 in previous sessions and is currently studying k_4 and k_5, the absence of a memory gate may prevent the model from recognizing the forgotten state of k_1, missing the opportunity for timely review. By introducing the memory gate, the model can assign greater weight to k_1 in the historical state, reflecting it in the prediction results and supporting timely review.

In this study, inspired by the LSTM gate mechanism, a memory gate was constructed for the GNN model, with its structure shown in Fig. 5. The forgetting vectors of knowledge concept k_i are first combined with the embedding vectors at times $t-1$ and t, producing the embedded hidden vectors $\mathbf{z}k_i{}^{t-1}$ and $\lambda k_i{}^{t-1}$ after applying forgetting processing at these times. Then, $\mathbf{z}k_i{}^{t}$ and $\lambda k_i{}^{t}$ are input into the memory gate to generate the embedding of the forgetting vectors $\mathbf{m}_{k_i}^{t'}$ for k_i, as shown in Eq. (8):

$$\mathbf{m}_{k_i}^{t'} = \tanh(\mathbf{W}_2(\mathbf{z}_{k_i}^{t-1} + \lambda_{k_i}^{t-1}) \otimes (\mathbf{z}_{k_i}^{t} + \lambda_{k_i}^{t}) + \mathbf{b}_2), \tag{8}$$

where $\mathbf{W}_2$ is the weight matrix and $\mathbf{b}_2$ is the bias term.

4.3.3.2. Design of the information propagation function. The information propagation function updates the representation of each node based on the aggregated features of its neighbors. In this study, two propagation functions are defined: f_{self} and $f_{neighbor}$. The function f_{self} handles information propagation for a knowledge concept k_i that belongs to C^t, considering only the forgetting feature of k_i and ignoring its knowledge graph features. The function $f_{neighbor}$ propagates information from a knowledge concept k_i to its neighbor nodes k_j, incorporating both the forgetting vector $\lambda_{k_i}^t$ and the adjacency matrix $\mathbf{AB}$ to update the embeddings of neighboring nodes. Eq. (9) defines these information propagation functions:

$$\begin{cases} f_{self}(\mathbf{m}_{k_i}^{t'}) = \mathbf{W}_2\mathbf{m}_{k_i}^{t'} + \mathbf{b}_2, & k_i \in C^t \\ f_{neighbor}(\mathbf{z}_{k_i}^{t'}, \mathbf{z}_{k_j}^{t'}) = \mathbf{AB}_{k_i,k_j} f_1([\mathbf{z}_{k_i}^{t'}, \mathbf{z}_{k_j}^{t'}]) + \mathbf{AB}_{k_j,k_i} f_2([\mathbf{z}_{k_i}^{t'}, \mathbf{z}_{k_j}^{t'}]) + f_3(\lambda_{k_i}^{t}), & k_i \notin C^t \end{cases}, \tag{9}$$

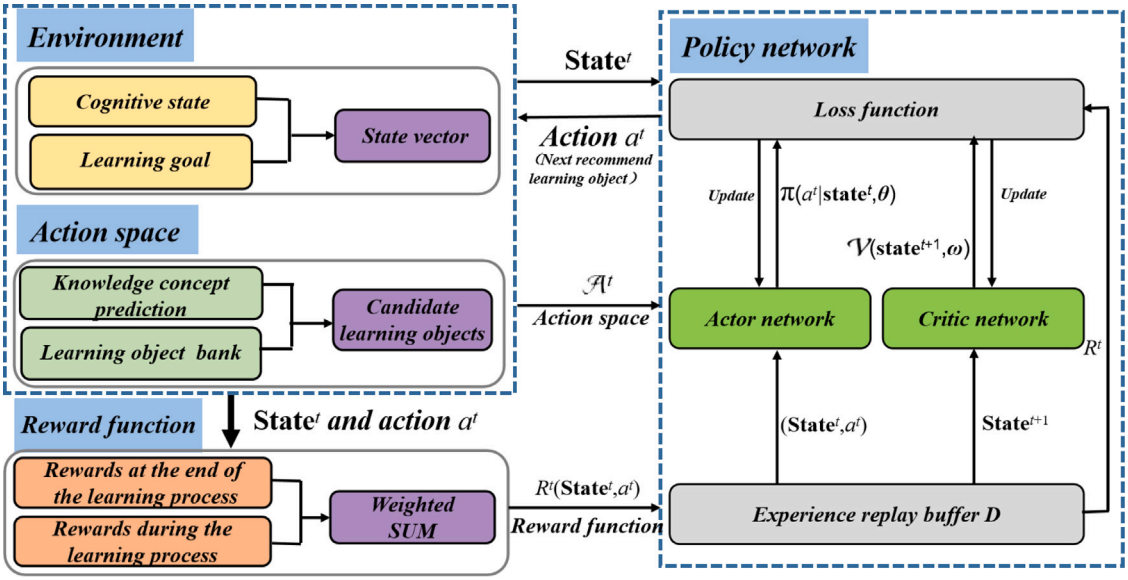


Fig. 6. Internal architecture of the reinforcement learning module.

where f_{self}, f_1, f_2 and f_3 are MLP.

The embedding vector of a learner's knowledge concept at moment $t+1$ is computed as shown in Eq. (10):

$$\mathbf{z}_{k_i}^{t+1} = \begin{cases} f_{self}(\mathbf{m}_{k_i}^{t'}) & k_i \in C^t \\ f_{neighbor}(\mathbf{z}_{k_i}^{t'}, \mathbf{z}_{k_j}^{t'}) & k_i \notin C^t. \end{cases} \tag{10}$$

4.3.4. Prediction layer

The prediction layer takes the knowledge concept embedding vector $\mathbf{z}_{k_i}^{t+1}$ as input to the MLP and produces the probability of each knowledge concept occurring at moment $t+1$. This output is represented by the knowledge concept coverage vector $\mathbf{p}^{t+1}$, calculated using Eq. (11):

$$\mathbf{p}^{t+1} = sigmoid(\mathbf{W}_{out}\mathbf{z}_{k_i}^{t+1} + \mathbf{b}_{out}). \tag{11}$$

Since knowledge concept prediction is formulated as a binary classification task, the loss function is calculated using the negative log-likelihood, as defined in Eq. (12):

$$\mathcal{L} = NLLLoss(([1-\mathbf{p}^{t+1}, \mathbf{p}^{t+1}]), \mathbf{p}^t), \tag{12}$$

where $\mathbf{p}^{t+1}$ represents the probability of a knowledge concept occurring, and $[1-\mathbf{p}^{t+1}, \mathbf{p}^{t+1}]$ provides a binary categorical representation indicating the presence or absence of the knowledge concept, where 0 denotes absence and 1 denotes presence.

4.4. Decision-making module

Since learning paths include both new and previously learned knowledge concepts that require a coherent learning sequence, a decision-making module is incorporated, and RL is employed to recommend learning paths. Fig. 6 presents the internal structure of the RL framework. The learner's cognitive state and learning goals are treated as the "environment state", while relevant learning objects are filtered from the LOB to form the "action space" based on predicted knowledge concepts. A reward function is defined for both the learning process and the completion phase, enabling the policy network to be trained to recommend optimal learning paths.

4.4.1. Environmental state

The learning environment evolves as learners interact with learning objects. In this study, the environmental state is defined as $\mathbf{state}^t = [s_i^t, lo_j^t, target, \mathbf{c}^t]$, where s_i^t and lo_j^t represent the learner and the learning object at time t, respectively. The feature values of these variables are directly obtained from the learning environment. The term $target$ denotes the learning goal set by each learner before starting the learning process, which may differ among learners. $\mathbf{c}^t$ indicates the learner's mastery of knowledge concepts at time t, calculated according to the method outlined in Section 4.2.

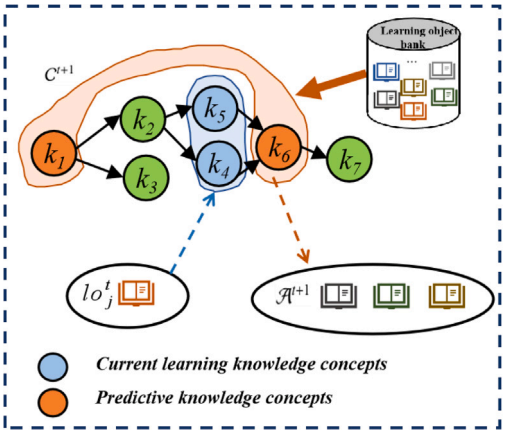


Fig. 7. Process of generating the action space based on predicted knowledge concepts.

4.4.2. Action space

Fig. 7 illustrates the process of generating the action space. Using the knowledge concept prediction results, the specific knowledge concepts that the learning object should include at the next step are computed via Eq. (13) and stored in the set C^{t+1}. For example, in Fig. 7, $C^{t+1} = k_1, k_6$.

$$C^{t+1} = \{k_i \mid p_{k_i}^{t+1} \geq 0.7\}, \tag{13}$$

Based on the knowledge concepts contained in the set C^{t+1}, the corresponding learning objects are selected from the learning resource library $\mathcal{LOB}$ to construct the candidate set of learning objects, denoted as $\mathcal{A}^{t+1}$. The computation is expressed in Eq. (14):

$$\mathcal{A}^{t+1} = \{lo_j^{t+1} \mid lo_j^{t+1} \in \mathcal{LOB}, k_i \in C^{t+1}, \Omega_{lo_j} \leq \Omega_{\mathrm{TOP_N}}\}, \tag{14}$$

where lo_j^{t+1} represents the candidate learning objects at time $t+1$, and $\Omega_{\mathrm{TOP_N}}$ denotes the top N eligible learning objects in the set C^{t+1}. The candidate set of learning objects generated at each step serves as the action space for the RL process, supporting the execution of subsequent recommendation tasks.

4.4.3. Reward function

The reward function is a central component for optimizing the agent's policy. It enables the agent to quantify task achievement and guides action selection through goal-directed feedback. During exploration, the reward provides positive or negative feedback for the agent's decisions, allowing its policy to be progressively adjusted and optimized. Additionally, the reward helps the agent balance exploration and exploitation, thereby promoting the formation of a long-term optimal policy.

Conventional RL-based adaptive learning path recommendation frameworks assign rewards only at the end of exploration. This delayed and sparse reward structure can lead to arbitrary choices during early exploration and result in unstable performance. Drawing on the concept of reward shaping (Zhang, Weng, Zhou, Zhang, & Huang, 2022), this study enhances the reward function by assigning rewards both at each step and at the end of the learning process. This approach ensures the effectiveness of the adaptive learning path and improves the agent's stability during exploration. The return function is defined in Eq. (15):

$$R^t = \alpha^* R_{step}^t + \beta^* R_{final}^t, \tag{15}$$

where R_{final}^t denotes the reward obtained at the end of the learning process, and R_{step}^t denotes the reward received at each step during the agent's exploration. During exploration, setting $\alpha = 1$, $\beta = 0$, and $R^t = R_{step}^t$, representing the reward obtained at each exploration step. Upon completion of the exploration, setting $\alpha = 0$ and $\beta = 1$ results in $R^t = R_{final}^t$, representing the reward for achieving the target knowledge concept.

4.4.3.1. End-of-learning process rewards. In industrial and gaming contexts, researchers often assign fixed values (e.g., 0 or 1) to guide agents in developing an optimal policy. However, such an approach is unsuitable for complex educational scenarios, where online learning environments must account for the alignment between learners and learning resources. Following the approach in Liu et al. (2019), the reward for learning outcomes is defined based on the improvement in learners' performance throughout the learning process, calculated as shown in Eq. (16):

$$R_{final}^t = \sum_t^{|N|} \gamma^t \left(\frac{E_e - E_s}{E_{sup} - E_s} \right)^t, \tag{16}$$

where E_s denotes the learner's initial score, E_e represents the score at the end of the learning process, E_{sup} is the total possible score of all learning objects, and γ is the discount factor, typically set to 0.99.

4.4.3.2. Rewards during the learning process. Rewards for the agent are defined based on the difficulty of each learning object during the exploration phase. Since a learning object's difficulty reflects the cognitive load and challenge experienced by learners, the reward signal captures the underlying effort and educational value, rather than merely indicating task completion. When the agent selects a learning object whose difficulty closely matches the learner's expected level, a high reward is assigned, calculated as shown in Eq. (17):

$$R^t_{step} = 1 - \mid \delta - dif^t_{lo_j} \mid, \tag{17}$$

where δ denotes the learner's expected difficulty level, and $dif^t_{lo_j}$ denotes the difficulty of the learning object lo_j at time step t.

4.4.4. Policy network

The Actor–Critic algorithm in RL is employed to generate adaptive learning path recommendations. The algorithm comprises two neural networks: the policy network (Actor) and the value network (Critic). The policy network is responsible for selecting learning objects from the candidate set based on the learner's current state, while the value network evaluates the quality of the selected items. Through their dynamic interaction, the policy network is trained. Once trained, the policy network retains the learning objects that yield the highest expected cumulative reward R^t for each learning state, as detailed in Algorithm 1.

In summary, the environment state continuously monitors changes in the learner's cognitive state, the action space generates candidate learning objects based on this state, and the reward function measures the match between the learner's cognitive state and the difficulty of the recommended items. By receiving feedback from the environment and updating its parameters via a policy optimization algorithm, the policy network iteratively refines the recommendation strategy, ultimately establishing a mechanism for learning path recommendations that optimizes long-term learning outcomes.

Algorithm 1: Actor–Critic training algorithm.

Input : Obtain the subject knowledge graph
Initialize policy network π, Critic network V, total number of training episodes T, experience replay buffer **D**;
for *episode* $\leftarrow 1$ **to** T **do**
  $\mathbf{state}^t, done^t \leftarrow reset()$;
  while $done^t$ *is* $false$ **do**
    Calculate $\mathcal{A}^{t+1}$ // by Equation (14);
    r randomly;
    if $r < \epsilon$ **then**
      $a^t \leftarrow$ random action from $\mathcal{A}^{t+1}$;
    else
      $a^t \leftarrow \pi(\mathbf{state}^t, \theta, \mathcal{A}^{t+1})$;
    end
    $\mathbf{state}^{t+1}, R^{t+1}, done^{t+1} \leftarrow step(a^t)$// R^{t+1} Calculate by Equation (16);
    Put $\{\mathbf{state}^t, a^t, \mathbf{state}^{t+1}, R^{t+1}\}$ into **D**;
    Sample batch $\{\mathbf{state}^t, a^t, \mathbf{state}^{t+1}, R^{t+1}\}$ from **D**;
    for *each sample in batch* **do**
      $\mathcal{L}^t \leftarrow R^{t+1} + \gamma \mathcal{V}\left(\mathbf{state}^{t+1}, w\right) - \mathcal{V}\left(\mathbf{state}^t, w\right)$ // The TD-ERROR loss ;
      Update Critic network: $w \leftarrow w + \mu_w \mathcal{L} \nabla_w V(\mathbf{state}^t, w)$;
      Update policy network: $\theta \leftarrow \theta + \mu_\theta \mathcal{L} \nabla_\theta \ln \pi(a^t | \mathbf{state}^t, \theta)$;
    end
    $t \leftarrow t + 1$;
  end
end
Output: Trained policy network π

5. Experiments

The evaluation of the FKGRec framework was conducted in four stages. First, validation experiments were performed using both a public dataset and a real-world dataset. Second, the framework was compared against state-of-the-art baseline methods. Third, the outcomes of these comparison experiments were analyzed in detail. Finally, case studies were conducted to further illustrate and support the experimental results.

5.1. Experimental settings

5.1.1. Datasets and data processing

The ASSISTments0910 dataset is a publicly available dataset from the ASSISTments 2010 system. It contains information such as learner ID, learning object ID, knowledge concept ID, the mapping between learning objects and knowledge concepts, and

Table 2
Summary of detailed information for the three datasets.

Dataset	Concepts	Students	Exercises	Records
ASSISTments0910	43	2259	8020	198,979
Adaptive learning system	132	266	470	42,401
Junyi	184	50,732	240	4,198,115

Table 3
Computation of feature values.

Feature dimension		ASSISTments0910 dataset	Adaptive learning system dataset	Junyi
Knowledge graph features	Knowledge concepts	*skill_name*	*kn_id*	*exercise*
	Relationships between knowledge concepts	Manually construct the subject knowledge graph; map learning objects to the graph; identify relationships; normalize	Load graph from the system; map learning objects to the graph; identify relationships; normalize	Load graph from the system; map learning objects to the graph; identify relationships; normalize
Forgetting features	Time intervals for the same knowledge concept	Compute time difference between the *problem_start_time* of the same concept; normalize.	Compute time difference between *start_ans_time* of the same knowledge concept; normalize	Compute the time difference between *time_done* of the same concept; normalize
	Sequential time intervals	Compute the difference between *problem_start_time* of the current knowledge concept and the *problem_end_time* of the previous knowledge concept; normalize	Compute difference between *start_ans_time* of the current knowledge concept and *created_at* of the previous knowledge concept; normalize	Compute difference between *start_ans_time* of the current knowledge concept and *time_taken* of the previous knowledge concept; normalize
	Number of times learned	Count learning objects corresponding to the same knowledge concept for each learner; normalize	Count learning objects corresponding to the same concept for each learner; normalize	Count learning objects corresponding to the same concept for each learner; normalize
Difficulty features		Computed according to Eq. (18)		

timestamps. For this study, the dataset was preprocessed to focus on mathematics, removing records lacking knowledge concepts or those with fewer than 10 learner interactions. As the ASSISTments system does not provide a built-in knowledge graph, two experienced mathematics teachers manually constructed one for this experiment. Following data cleaning and alignment to reflect the logical relationships among knowledge concepts, the final dataset comprised 43 knowledge concepts, 2259 learners, 198,979 interaction records, and 8020 learning objects.

The adaptive learning system dataset is sourced from our team's self-developed adaptive learning system,[1] which integrates a subject-specific knowledge graph during its development. For this study, data from the first three chapters of the C programming course were selected. The dataset was processed using the same cleaning procedure applied to the ASSISTments0910 dataset, yielding 132 knowledge concepts, 266 learners, 42,401 interaction records, and 470 learning objects.

The Junyi dataset is publicly available[2]and includes logs from around 250,000 students with more than 25 million interactions. For this study, only algebra-related data were extracted. The dataset was processed using the same cleaning procedure applied to the ASSISTments0910 dataset, resulting in 184 knowledge concepts, 50,732 learners, 4,198,115 interaction records, and 240 learning objects. Table 2 presents a summary of the cleaned data across the three datasets.

The extraction of knowledge graph features, forgetting features, and difficulty features is a necessary step for implementing the FKGRec framework. These features can be obtained by preprocessing the corresponding fields within the datasets, as summarized in Table 3. While knowledge graph and forgetting features can be computed through relatively straightforward procedures, estimating difficulty features is more challenging, since neither dataset provides explicit difficulty labels. Following the method proposed in the literature (Ren et al., 2023), difficulty values are therefore approximated using a calculation outlined in Eq. (18).

$$dif_{lo_j}^{t} = \frac{\sum_{i=1}^{|\mathrm{K}|}(correct_i \mid lo_j^{k_i} == 1)}{\sum_{j=1}^{|\mathrm{K}|} lo_j^{k_i}}, \tag{18}$$

where |K| represents the total number of knowledge concepts in a course, $lo_j^{k_i}$ refers to the ith knowledge concept associated with the learning object lo_j, and $correct_i$ indicates the accuracy of the ith knowledge concept.

5.1.2. Baseline frameworks

The proposed FKGRec framework is evaluated against the following baseline approaches:

[1] http://120.53.238.22.
[2] https://www.junyiacademy.org.

- **KU-CF** (Mu & Yuan, 2024): This framework integrates collaborative filtering with a knowledge graph. It utilizes semantic information from the learner–item interaction matrix in conjunction with knowledge concepts to calculate similarities between learning objects, thereby facilitating learning path recommendations.
- **LSTMPR** (Zhou et al., 2018): This method first clusters learners based on their characteristics and then applies an LSTM model to predict their learning paths. Personalized learning paths are subsequently derived from these predictions.
- **SASRec** (Kang & McAuley, 2018): SASRec leverages a stacked multi-head self-attention mechanism combined with feedforward neural networks to model student exercise sequences. A single-layer Transformer is then employed to generate learning path recommendations.
- **KCP-ER** (Ma et al., 2022): KCP-ER uses deep knowledge tracing to infer learners' cognitive states from historical learning records. A recurrent neural network then predicts the knowledge concepts each learner should study at each step, and exercises corresponding to these concepts are selected for recommendation.
- **MulOER-SAN** (Ren et al., 2023): This two-layer, multi-objective recommendation framework employs an attention mechanism to track learners' cognitive states and generates sets of exercises with moderate difficulty for each recommendation step. The chaotic sparrow search algorithm is subsequently applied to refine the exercise set and produce personalized learning paths.
- **CSEAL** (Liu et al., 2019): CSEAL combines RL with LSTM-based cognitive diagnosis and a knowledge graph for knowledge concept navigation. RL algorithms are then used to recommend tailored learning paths.

5.1.3. Parameter settings

The parameter configurations for the baseline frameworks follow the specifications reported in their original studies. For the FKGRec framework, the LSTM model uses a feature dimension equal to the number of knowledge concepts, with a hidden layer size of 32. In the MemGNN method, both the hidden layer and the embedding layer sizes are set to 32. The training process is conducted over 32 epochs with a batch size of 32. In the Actor–Critic model, each session concludes when the learner either completes the learning objective or reaches the maximum allowable learning duration. The Adam optimizer is applied across all models, with an initial learning rate of 0.01 and a decay factor of 0.5 every 200 steps. Dropout is set to 0.5 to mitigate overfitting, and the negative log-likelihood loss function is employed. All DL models are trained using 80% of the dataset for training, 10% for validation, and 10% for testing, with experiments conducted using random data selection.

5.1.4. Evaluation metrics

The performance of the FKGRec framework is evaluated using four metrics: **accuracy** (Ma et al., 2024), **effectiveness** (Liu et al., 2019), **adaptivity** (Ren et al., 2023), and **diversity**.

(1) **Accuracy**

Accuracy is employed to assess the correctness of node ordering within the recommended learning path sequence. In this study, precision, recall, and the F1 score are adopted to comprehensively evaluate this metric. The corresponding formulas are defined as follows:

$$Precision(\mathcal{LP}) = \frac{|LCS(\mathcal{LP}_{pred}, \mathcal{LP}_{actual})|}{|\mathcal{LP}_{pred}|}, \tag{19}$$

$$Recall(\mathcal{LP}) = \frac{|LCS(\mathcal{LP}_{pred}, \mathcal{LP}_{actual})|}{|\mathcal{LP}_{actual}|}, \tag{20}$$

$$F1(\mathcal{LP}) = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{21}$$

where LCS denotes the length of the Longest Common Subsequence between two sequences. $\mathcal{LP}pred$ refers to the predicted learning path sequence, $\mathcal{LP}actual$ indicates the actual learning path sequence, and $|\cdot|$ represents the sequence length.

(2) **Effectiveness**

Effectiveness serves as a key metric for evaluating the outcomes of a learner's learning process. Its purpose is to measure the degree of improvement in the learner's performance over the course of the learning process. It is expressed by Eq. (22):

$$E_p(\mathcal{LP}) = \frac{E_e - E_s}{E_{\text{sup}} - E_s}, \tag{22}$$

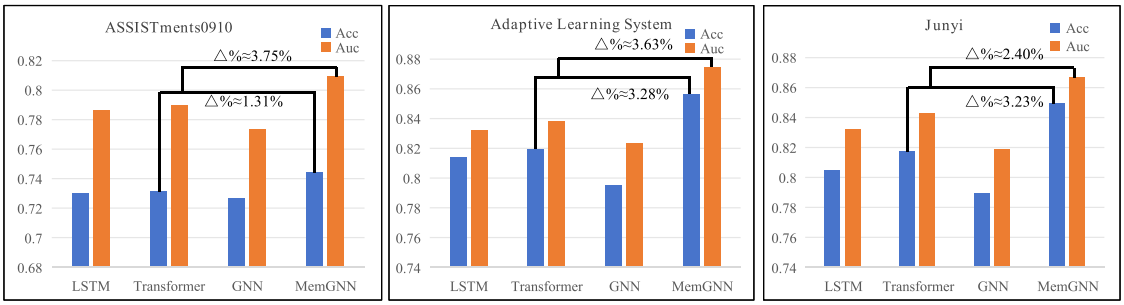
where E_s represents the learner's initial score, E_e denotes the final score, and E_{sup} indicates the total attainable score across all learning objects. $E_p(\mathcal{LP})$ reflects the magnitude of performance improvement throughout the learning process.

(3) **Adaptivity**

Adaptivity evaluates the extent to which the recommended learning objects correspond to the learner's current cognitive level. Its purpose is to determine how effectively the difficulty of each learning object aligns with the learner's cognitive state. The metric is formulated in Eq. (23):

$$Adaptivity(\mathcal{LP}) = \frac{\sum_{i=1}^{|\mathrm{M}|}(1 - |\,\delta - dif_{lo_j}^{t}\,|)}{|\,\mathrm{M}\,|}, \tag{23}$$

where δ denotes the learner's target difficulty level, $dif_{lo_j}^{t}$ represents the actual difficulty of the learning object lo_j at time t, and M is the total length of the learning path.



(a) Comparative experimental results of ASSISTments0910 of different methods

(b) Comparative experimental results of Adaptive Learning System of different methods

(c) Comparative experimental results of Junyi of different methods

Fig. 8. Results of the knowledge concept prediction methods.

(4) **Diversity**

Diversity evaluates the degree of repetitiveness among learning objects recommended within a learning path. Frequent recurrence of identical learning objects may result in learner disengagement and diminished interest. The metric is formulated as shown in Eq. (24):

$$Diversity(\mathcal{LP}) = \frac{\sum_{lo_i \in \mathcal{LP}} \sum_{lo_j \in \mathcal{LP}} (1 - \mathrm{sim}(lo_i, lo_j))}{|\,\mathrm{M}\,||\,\mathrm{M} - 1\,|}, \tag{24}$$

lo_i and lo_j denote the learning objects within a learning path, $\mathrm{sim}(lo_i, lo_j)$ represents the cosine similarity between them, and M indicates the length of the learning path list.

5.2. Experimental results

5.2.1. Experimental results of the knowledge concept prediction methods

To evaluate the accuracy of MemGNN predictions and examine whether the predicted knowledge concepts at each recommendation step encompass both new and previously learned concepts, we first compare the performance of MemGNN with other knowledge concept prediction methods. Subsequently, the counts of new and previously learned knowledge concepts included in the predictions at each step are analyzed. Finally, ablation experiments are conducted to investigate the influence of forgetting factors.

5.2.1.1. MemGNN performance comparison. Fig. 8 presents the experimental results of the MemGNN method in comparison with three baseline approaches – LSTM (Wu, Li, Tang, & Liang, 2020), Transformer (Ren et al., 2023), and GNN – across three datasets. Two widely used evaluation metrics are employed: Accuracy (ACC) and the area under the ROC curve (AUC), which quantify the proportion of correctly predicted samples and aid in assessing model performance. In the figure, Δ denotes the performance gap between the best and the second-best results for each dataset on both evaluation metrics. Based on Fig. 8, the following conclusions can be drawn:

(1) The MemGNN method surpasses all other approaches across all evaluation metrics. Specifically, on the ASSISTments0910 dataset, the average improvements in ACC and AUC are 1.31% and 3.75%, respectively. On the adaptive learning system dataset, the corresponding improvements are 3.28% and 3.63%, while on the Junyi dataset, they are 3.23% and 2.40%. These findings confirm that MemGNN enhances the accuracy of knowledge concept prediction.

(2) A comparison of the LSTM and Transformer methods shows that the Transformer consistently outperforms the LSTM across all three datasets in both ACC and AUC metrics, indicating its superior capability in predicting knowledge concept coverage.

(3) A comparison between the GNN and MemGNN methods indicates that integrating forgetting features enhances algorithmic performance, suggesting that such incorporation improves the accuracy of knowledge concept prediction.

(4) A comparison of the GNN, LSTM, and Transformer models shows that the GNN consistently underperforms relative to the other two across all three datasets. This outcome may be attributed to the fact that the LSTM and Transformer models predict both new and previously learned knowledge concepts, whereas the GNN model predicts only new concepts.

(5) The ACC achieved with the adaptive learning system and Junyi datasets is higher than that of the ASSISTments0910 dataset. This disparity arises because the ASSISTments0910 dataset originally lacked a knowledge graph, relying instead on a conventional "topic–knowledge concept" mapping. Although two experienced teachers later constructed a knowledge graph for this dataset, its initial structure was not fully aligned with graph representation, constraining the model's ability to capture inter-concept relationships and reducing prediction accuracy. By contrast, the adaptive learning system and Junyi datasets were developed with embedded knowledge graphs, ensuring that their interaction data were inherently graph-based. This design enabled the MemGNN model to more effectively extract graph-structured information, integrate learners' forgetting features, and thereby enhance prediction performance.

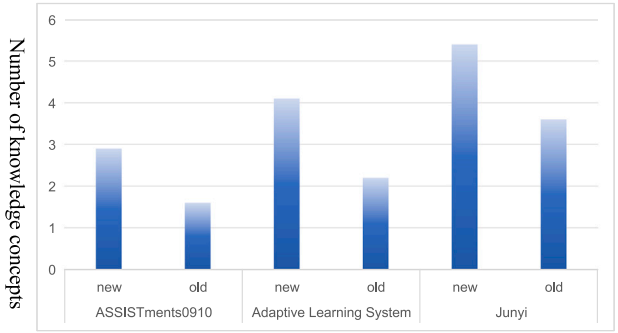


Fig. 9. Counts of predicted knowledge concepts.

Table 4
Ablation experiment results.

Model	Acc		
	ASSISTments0910	Adaptive learning system	Junyi
MemGNN w/o MG	0.7292	0.7987	0.7856
MemGNN w/o IPF	0.7305	0.8171	0.8114
MemGNN w/o Δt	0.7348	0.8355	0.8466
MemGNN w/o Δs	0.7319	0.8325	0.8258
MemGNN w/o Δr	0.7353	0.8334	0.8352
MemGNN	**0.7443**	**0.8568**	**0.8497**

5.2.1.2. Knowledge concept prediction analysis. Fig. 9 presents the average number of new and previously learned knowledge concepts predicted by FKGRec at each step across both datasets. The results demonstrate that FKGRec effectively captures forgotten concepts. Consequently, its predictions expand the coverage of knowledge concepts within the action space, enabling more comprehensive recommendations that encompass both newly introduced and previously acquired concepts.

5.2.1.3. Ablation experiments. To examine the influence of the core components of MemGNN and each forgetting factor on average ACC, a series of ablation experiments were conducted across three datasets to identify the key contributors to the model's overall performance. Since MemGNN extends the GNN model by incorporating a memory gate structure and redesigning the information propagation function, the results in Fig. 8 indicate that MemGNN outperforms GNN in knowledge concept prediction. This highlights the importance of both the memory gate and the information propagation function. Accordingly, this study evaluates their individual contributions to average ACC. In Table 4, "MemGNN w/o MG" denotes the variant without the memory gate, and "MemGNN w/o IPF" represents the variant without the information propagation function. With respect to forgetting factors, "MemGNN w/o Δt" indicates the model variant without Δt, while "MemGNN w/o Δs" and "MemGNN w/o Δr" refer to variants excluding Δs and Δr, respectively. The results in Table 4 form the basis for the following conclusions.

(1) Within the core components of the MemGNN model, both MemGNN w/o MG and MemGNN w/o IPF exhibit substantial performance degradation compared to the original MemGNN. These results demonstrate that the incorporation of the memory gate structure and the information propagation function contributes positively to improving ACC. Specifically, the memory gate enables the model to capture learners' forgetting characteristics during the learning process, while the information propagation function facilitates the effective integration of forgetting features with knowledge graph features. Collectively, these components strengthen the model's capability to accurately identify the knowledge concepts that learners should study at each step.
(2) Regarding the forgetting factors, the MemGNN model consistently outperforms its variants with any single factor removed across all three datasets. These findings indicate that each of the three forgetting factors contributes positively to the model's performance, and eliminating any one of them leads to a decline in accuracy. Among the variants, MemGNN w/o Δs exhibits the greatest performance loss, highlighting the substantial influence of Δs (the time interval between knowledge concept sequences) on ACC. In contrast, MemGNN w/o Δt and MemGNN w/o Δr show comparatively smaller performance declines, suggesting that Δt (the time interval of the same knowledge concept) and Δr (the frequency of the same knowledge concept) also affect ACC, but to a lesser extent than Δs.

5.2.2. Comparative experimental results of the FKGRec framework

To assess whether constructing the action space from predicted knowledge concepts enhances the quality of learning path recommendations, the FKGRec framework was evaluated against baseline models. Furthermore, a case study was conducted on the learning paths generated by FKGRec to examine the sequencing of newly introduced and previously learned knowledge concepts.

Table 5
Comparison of experimental accuracy results.

Frameworks	ASSISTments0910			Adaptive learning system			Junyi		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
KG-CF	0.152	0.227	0.182	0.336	0.391	0.361	0.271	0.320	0.293
LSTMPR	0.211	0.189	0.199	0.383	0.365	0.374	0.322	0.284	0.302
SASRec	0.125	0.256	0.168	0.460	0.414	0.436	0.239	0.358	0.287
KCP-ER	0.189	0.166	0.177	0.436	0.397	0.416	0.253	0.357	0.296
MulOER-SAN	0.230	0.206	0.217	0.453	0.451	0.452	0.454	0.446	0.450
CSEAL	0.226	0.274	0.248	0.428	0.413	0.420	0.457	0.415	0.435
FKGRec (ours)	**0.316**	**0.345**	**0.330**	**0.572**	**0.562**	**0.567**	**0.575**	**0.550**	**0.562**
Δ%	8.6%	7.1%	8.2%	11.9%	11.1%	11.5%	11.8%	10.4%	11.2%

Table 6
Comparison of experimental results for effectiveness, adaptivity, and diversity.

Frameworks	ASSISTments0910			Adaptive learning system			Junyi		
	Effectiveness	Adaptivity	Diversity	Effectiveness	Adaptivity	Diversity	Effectiveness	Adaptivity	Diversity
KG-CF	0.305	0.582	0.684	0.327	0.611	0.679	0.349	0.604	0.604
LSTMPR	0.382	0.604	0.597	0.399	0.631	0.643	0.395	0.647	0.647
SASRec	0.428	0.695	0.699	0.418	0.753	0.792	0.432	0.749	0.749
KCP-ER	0.457	0.757	0.741	0.461	0.796	0.848	0.446	0.778	0.778
MulOER-SAN	0.471	0.769	0.746	0.494	0.809	0.851	0.501	0.813	0.854
CSEAL	0.432	0.731	0.757	0.456	0.763	0.867	0.454	0.767	0.859
FKGRec (ours)	**0.518**	**0.796**	**0.768**	**0.533**	**0.832**	**0.872**	**0.530**	**0.847**	**0.868**
Δ%	4.7%	2.7%	1.1%	3.9%	3.3%	0.5%	2.9%	3.4%	0.9%

Table 7
Comparison of the internal structures of the FKGRec and CSEAL frameworks.

Frameworks	Learning environment characterization	Action space	RL algorithm
CSEAL	Learning goal, Cognitive state	Forming an action space using the set of successor knowledge concepts to the current knowledge concept in the knowledge graph	Actor–Critic
FKGRec	Learning goal, Cognitive state	Forming an action space using knowledge concept prediction	Actor–Critic

5.2.2.1. Overall comparison. This section presents the comparison results of seven learning path recommendation frameworks evaluated on four metrics: accuracy, effectiveness, adaptivity, and diversity. The accuracy results are shown in Table 5, while Table 6 reports the results for effectiveness, adaptivity, and diversity. The best results are highlighted in bold, and the second-best are underlined.

Several key findings are outlined below:

(1) The FKGRec framework achieves the highest performance in accuracy across all datasets. It also demonstrates substantial improvements over the best-performing baselines in effectiveness and adaptivity, although its advantage in diversity is less pronounced. Specifically, in terms of effectiveness, FKGRec surpasses the best baseline (MulOER-SAN) by 3.9%–4.7%, indicating a stronger ability to enhance learning outcomes. In terms of adaptivity, it outperforms MulOER-SAN by 2.7%–3.3%, suggesting that the recommended learning paths are more aligned with learners' cognitive states. However, for diversity, FKGRec shows only marginal improvements (0.5%–1.1%) over the best baseline (CSEAL). This suggests that while RL–based approaches such as FKGRec and CSEAL are effective in improving diversity by expanding the range of recommended learning materials, FKGRec's diversity advantage is relatively modest.
An additional observation concerns the diversity metric of the MulOER-SAN framework on the ASSISTments0910 dataset, which differs significantly from results reported in Ref. Ren et al. (2023). This discrepancy may arise from differences in data preprocessing. Earlier work removed only incomplete records and students with fewer than three interactions, while the present study applied stricter cleaning by excluding students with fewer than ten interactions and constructing a knowledge graph to refine relationships among concepts. These changes likely influenced the diversity outcomes.

(2) A comparison between the FKGRec and CSEAL frameworks shows that FKGRec performs better in terms of both effectiveness and adaptability. To interpret these findings, the study examines the internal structures of both frameworks (Table 7). While CSEAL and FKGRec share similarities in characterizing the learning environment and applying RL algorithms, they differ in how the action space is designed. CSEAL constructs its action space from successor knowledge concepts within the knowledge graph, which does not account for forgotten concepts. By contrast, FKGRec generates its action space through knowledge concept prediction, enabling the inclusion of forgotten knowledge in the recommendations. Revisiting forgotten concepts not only strengthens learners' performance but also provides exercises of appropriate difficulty tailored to their cognitive state.

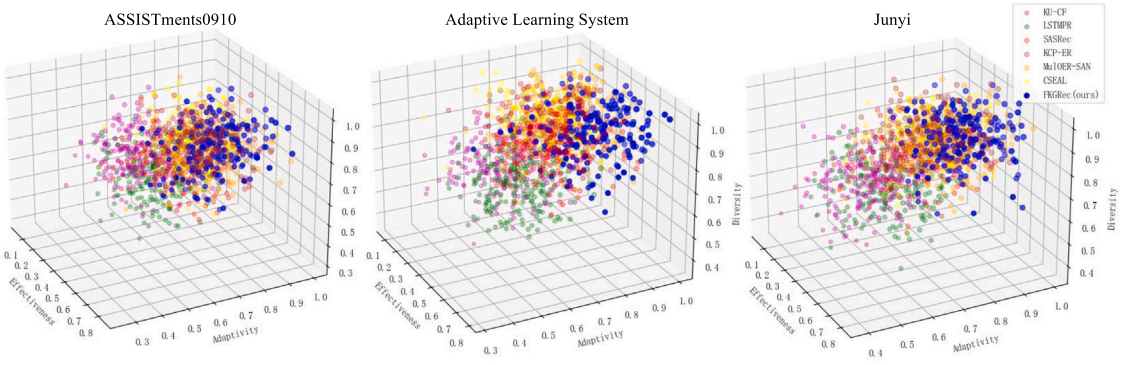


Fig. 10. Comparison of the balance among evaluation metrics.

Table 8
Ablation study of the FKGRec framework.

Framework	ASSISTments0910			Adaptive learning system			Junyi		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
FKGRec w/o CD	0.194	0.209	0.201	0.368	0.377	0.372	0.386	0.369	0.377
FKGRec w/o KCP	0.216	0.266	0.238	0.419	0.401	0.410	0.434	0.411	0.422
FKGRec w/o DM	0.201	0.257	0.226	0.389	0.395	0.392	0.416	0.398	0.407
FKGRec	0.316	0.345	0.330	0.572	0.562	0.567	0.575	0.550	0.562

5.2.2.2. Balance comparison. High effectiveness in learning paths does not necessarily imply high adaptivity or diversity. To evaluate the trade-off among these three dimensions, 100 learning paths were randomly sampled from the test sets of each framework, and their metric values were visualized. Fig. 10 presents the scatter plots, where the X-, Y-, and Z-axes denote adaptivity, effectiveness, and diversity, respectively. The results show that, across the three datasets, the FKGRec framework yields more balanced performance, with learning paths distributed in more optimal regions than those of the baseline models.

5.2.3. Ablation experiments of the FKGRec framework

To examine the contribution of each core module in the FKGRec framework to average precision, recall, and F1 score, a series of ablation studies were performed across three datasets. As outlined in Section 4.2, FKGRec consists of four main modules: the data preparation module, the cognitive diagnosis module, the knowledge concept prediction module, and the decision-making module. The data preparation module was excluded from ablation since its role is to process raw learning logs and generate input data that satisfy the framework's requirements. This function provides the foundation for all subsequent modules, and its removal would render the framework inoperable. Therefore, it was deemed unsuitable for ablation, as its focus is on data construction rather than algorithmic design.

For the remaining three modules, the following ablation settings were implemented: (1) Without the cognitive diagnosis module (FKGRec w/o CD). The cognitive diagnosis mechanism is replaced with a simple historical interaction frequency as the learner model. This setting evaluates the value of incorporating fine-grained cognitive diagnosis. (2) Without the knowledge concept prediction module (FKGRec w/o KCP). In this variant, the model does not predict the next knowledge concept the learner should study. Instead, it generates a candidate set using the knowledge graph and randomly selects a concept. This setting examines the influence of the knowledge concept prediction module on recommendation performance. (3) Without the decision-making module (FKGRec w/o DM). The RL strategy is substituted with a heuristic-based recommendation strategy. This experiment assesses the contribution of the adaptive decision-making mechanism.

As shown in Table 8, the complete FKGRec framework consistently outperforms all ablated variants across the three datasets, demonstrating that each core module plays a critical role in overall performance. Excluding any module leads to notable reductions in precision, recall, and F1 score. Collectively, the results validate the effectiveness of the three modules. The cognitive diagnosis module enhances recommendation quality through personalized cognitive modeling. The knowledge concept prediction module combines forgetting features and knowledge graph features to forecast both new and previously encountered knowledge concepts relevant to the learner's current state, thereby improving focus and accuracy. The decision-making module applies an RL-based optimization strategy to generate pedagogically appropriate learning paths that balance new and prior knowledge. By optimizing recommendations across multiple rounds, it maximizes long-term learning gains and enhances overall outcomes. Together, these three modules operate synergistically to improve the adaptive recommendation capabilities of the FKGRec framework.

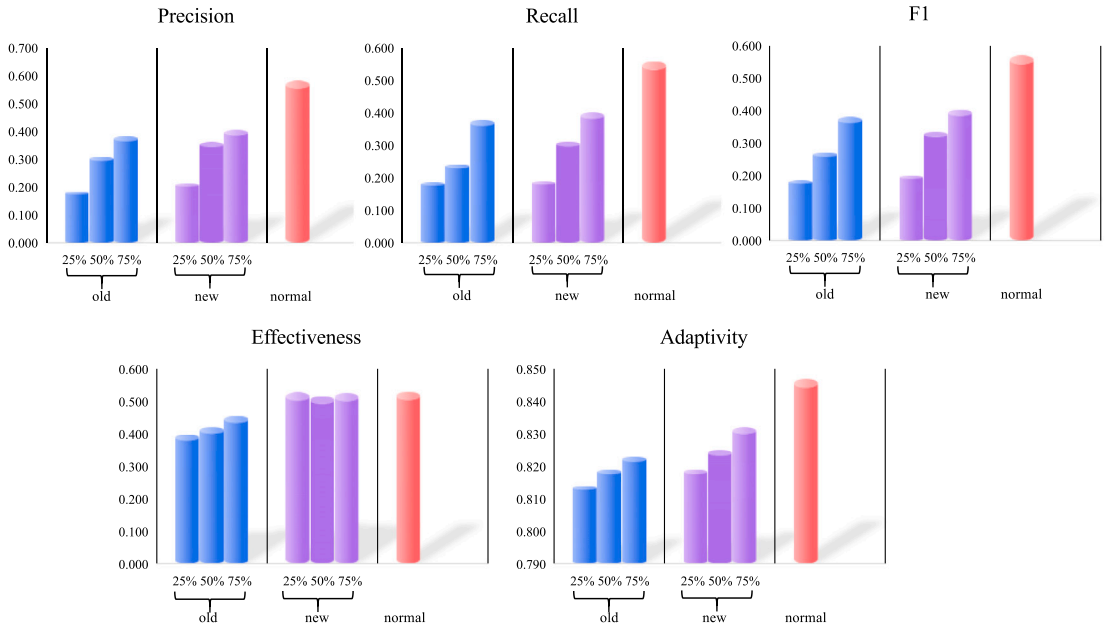


Fig. 11. Impact of the number of new and previously learned knowledge concepts on FKGRec framework performance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.2.4. Analysis of the number of new and old knowledge concepts

A series of experiments was conducted on the Junyi dataset to examine how the number of new and previously learned knowledge concepts in the action space influences the performance of the FKGRec framework. The performance was evaluated using precision, recall, F1 score, effectiveness, and adaptivity.

The results were illustrated in Fig. 11. The blue bars indicate performance when 100% of new knowledge concepts are included, combined with 25%, 50%, 75%, and 100% of old knowledge concepts. The purple bars show performance when 100% of old knowledge concepts are included, combined with 25%, 50%, and 75% of new knowledge concepts. The red bars represent the baseline, where 100% of both new and old knowledge concepts are included. Based on these results, two main conclusions can be drawn:

(1) Reducing the proportion of either new or old knowledge concepts leads to a decline in accuracy (precision, recall, F1) and adaptivity. The larger the reduction, the greater the performance degradation. This can be attributed to the fact that limiting the action space excludes potentially optimal knowledge concepts, with higher reductions increasing the likelihood of exclusion and, consequently, lowering accuracy and adaptivity.

(2) When the number of old knowledge concepts is reduced, effectiveness decreases, whereas reducing new knowledge concepts has little to no impact. This may be because a smaller set of old knowledge concepts hinders learners' ability to review and reinforce forgotten material, thereby lowering academic performance. In contrast, new knowledge concepts are generated from the knowledge graph and follow the logical progression of knowledge. Therefore, selecting any new knowledge concept supports knowledge construction and contributes positively to learning outcomes.

5.2.5. Parameter analysis

To examine the influence of model parameters on training performance, this study conducted parameter analysis from three aspects: embedded dimension, hidden dimension, and reward function hyperparameter settings.

5.2.5.1. Embedding dimension and hidden dimension. In this study, the embedding and hidden dimensions of the RL model were varied within the range 16, 32, 64, 128, 256 to assess their impact on recommendation performance. As shown in Fig. 12, the model achieved optimal results at an embedding dimension of 32. Performance improved when the dimension increased from 16 to 32, but declined at higher values. Similarly, Fig. 13 illustrates the effect of hidden layer dimensions. Performance improved as the hidden dimension increased from 16 to 32; however, further increases to 64, 128, and 256 led to consistent performance degradation, with a particularly sharp decline at 256.

This trend occurs because at low embedding and hidden dimensions, the RL model struggles to capture complex feature relationships within the data. Moderate increases enable the model to learn richer feature representations, thereby enhancing accuracy. However, excessively high dimensions cause overfitting, where the model captures noise and local patterns rather than generalizable features. In addition, larger dimensions increase the number of model parameters, raising training complexity and computational costs, which may result in instability and reduced performance.

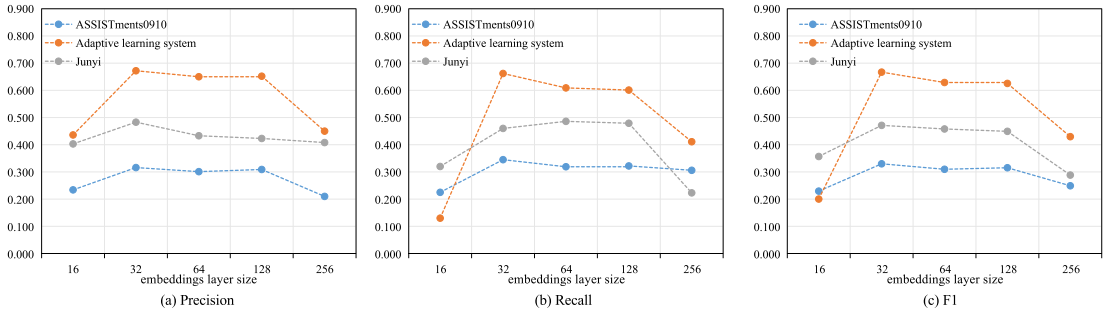


Fig. 12. Performance across different embedding dimensions on three datasets.

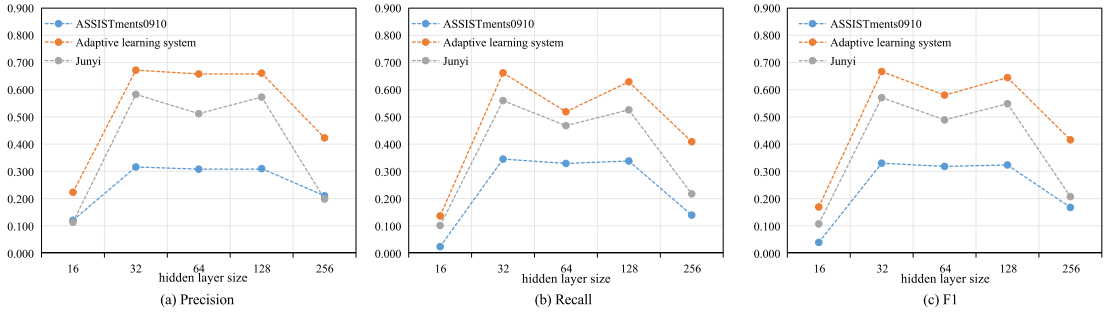


Fig. 13. Performance across different hidden dimensions on three datasets.

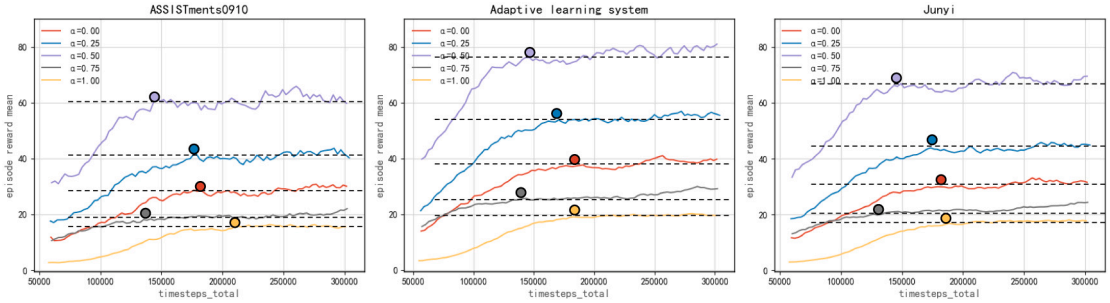


Fig. 14. Results under different hyperparameters across three datasets. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.2.5.2. Reward function hyperparameter setting. As defined in Eq. (15), the reward function consists of two stages: the learning process and the end of learning. Two hyperparameters, α and β, represent the weights of these stages. Fig. 14 shows the convergence behavior of the RL model under different α and β settings. The horizontal axis represents the total number of interactions between the recommender and the environment, while the vertical axis shows the average cumulative reward obtained when completing a learning path. Different colored curves correspond to different combinations of α and β.

Across all three datasets, the convergence speed follows the order: gray curve ($\alpha = 0.75, \beta = 0.25$), purple curve ($\alpha = 0.50, \beta = 0.50$), blue curve ($\alpha = 0.25, \beta = 0.75$), red curve ($\alpha = 0.00, \beta = 1.00$), and yellow curve ($\alpha = 1.00, \beta = 0.00$), with convergence speed increasing gradually. This finding indicates that assigning rewards to both the learning process and the final outcome accelerates convergence more effectively than rewarding only one stage. The fastest convergence occurs when $\alpha = 0.75, \beta = 0.25$, likely because frequent process-based rewards alleviate the sparse reward problem in RL, thereby speeding up convergence.

5.2.6. Running time analysis

To evaluate the computational efficiency in real-time recommendation tasks, the average time required to recommend each learning object during training was recorded. As shown in Fig. 15, the horizontal axis represents the number of interactions between the recommender and the environment, while the vertical axis shows the average time per recommendation (in ms). Results indicate that once training stabilizes, FKGRec requires approximately 10 ms more per recommendation than other frameworks. This overhead arises from incorporating both an RL learning mechanism and a memory gate structure within the graph neural network, which

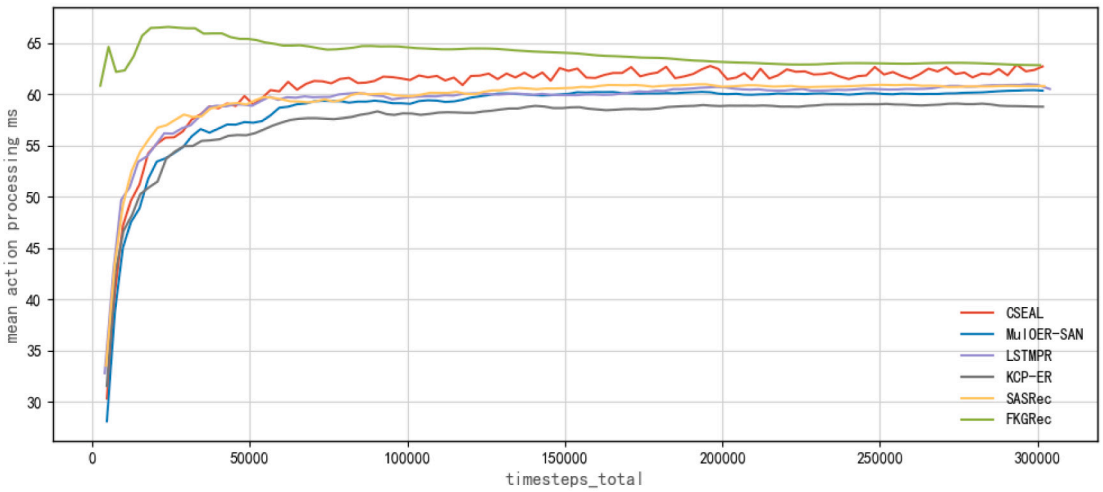


Fig. 15. Runtime analysis of different algorithms.

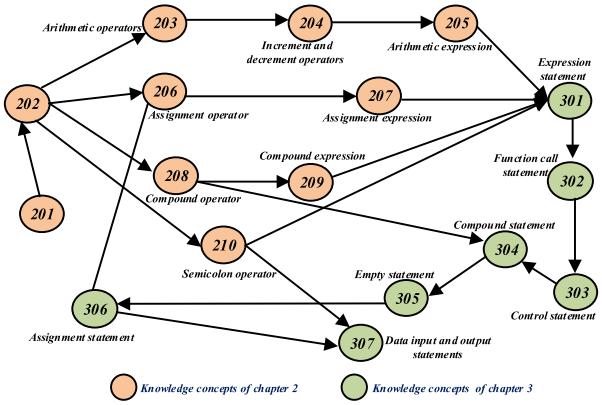


Fig. 16. Knowledge graph constructed from a subset of knowledge concepts.

necessitates more complex state computations and action selection. Despite this modest increase in runtime, FKGRec demonstrates superior performance in learning effectiveness, path rationality, and long-term outcomes. These strengths suggest that the framework remains both feasible and practical for real-time recommendation tasks in educational contexts.

5.2.7. Case study

To examine the learning order of new and previously forgotten knowledge concepts within the learning paths recommended by the FKGRec framework, a learner was randomly selected from the adaptive learning system dataset. Analysis of this learner's learning behavior pattern provides further insight into how the framework balances the recommendation of new knowledge with the review of forgotten concepts, thereby illustrating the principles that guide its learning path generation. This approach also allows an assessment of whether the FKGRec framework effectively adapts to the learners' needs for both exploration and revision.

This study selected a subset of knowledge concepts from the adaptive learning system dataset to construct the knowledge graph shown in Fig. 16. This graph was used to analyze the sequence of new and previously forgotten knowledge concepts recommended by the FKGRec framework for a specific learner. Since the learning objects recommended by the adaptive learning system often involve multiple concepts, displaying all of them would make it difficult to clearly track the learner's behavior patterns. To address this, the core concept of each learning object was extracted and visualized. As illustrated in Fig. 17, the learning path recommended for this learner is presented. Green arrows represent the order in which new knowledge concepts are recommended by the FKGRec framework; yellow arrows denote the order of forgotten concepts; purple arrows indicate repeated recommendations of the same concepts; and blue arrows mark cases where certain concepts are skipped. The thickness of the arrows reflects the frequency of each type of learning behavior.

From Fig. 17, several conclusions can be drawn:

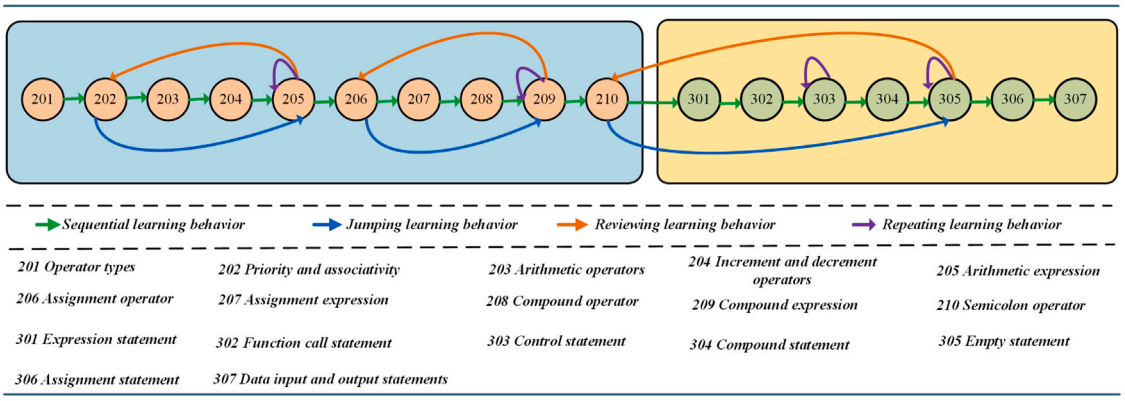


Fig. 17. Learning behavior pattern of a learner. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Conclusion 1: The new concepts recommended by the FKGRec framework follow the logical relationships among knowledge concepts, ensuring that the learning path progresses in a coherent and structured manner.

Conclusion 2: The framework reinforces retention by repeatedly recommending previously forgotten concepts, allowing learners to strengthen memory through review.

Conclusion 3: When learners show high mastery of certain concepts, the framework skips them and instead prioritizes unmastered content, tailoring recommendations to learners' cognitive states and current learning needs. These results highlight the adaptability of the framework in optimizing the learning process based on both new and reviewed content.

Building on this case analysis, this study further explores the applicability of the FKGRec framework across different review scenarios. In practice, review is often divided into daily review and exam preparation. Daily review emphasizes systematic mastery of knowledge, while also seeking timely support when learners encounter difficult concepts. Conclusions 1 and 2 indicate that the FKGRec framework can recommend content according to logical relationships and encourage both review of related concepts and repeated practice, making it well suited for daily review. In exam preparation scenarios, however, limited time leads learners to prioritize core and error-prone concepts. According to Conclusion 3, the framework can skip concepts that are already mastered and focus on identifying and addressing gaps. However, it is not yet capable of concentrating specifically on core concepts. Further refinements are therefore needed to improve its ability to support targeted review in exam-focused contexts.

6. Discussion

6.1. Results analysis

The FKGRec framework integrates a MemGNN method to predict knowledge concepts at each step of the learning process, followed by the application of an RL algorithm to recommend learning paths. Both MemGNN and FKGRec were evaluated against their respective baselines, and the experimental results demonstrate their superiority. Specifically, the prediction results for knowledge concept coverage show that MemGNN can simultaneously capture both newly acquired and forgotten knowledge concepts, achieving higher prediction accuracy than competing methods. Ablation studies further confirm that the information propagation function contributes significantly to performance improvements.

With respect to learning path recommendation, FKGRec substantially outperforms baseline frameworks in prediction accuracy, learner score improvement, difficulty alignment of learning objects, and maintaining learning interest. The ablation study results highlight that the cognitive diagnosis module, knowledge concept prediction module, and decision-making module each play a critical role in overall performance. Moreover, through hyperparameter optimization, the framework demonstrates strong performance across multiple datasets. Although its runtime does not present a clear advantage, comprehensive performance indicators suggest that this limitation does not reduce its practical applicability. Case analysis further shows that the sequencing of new and prior knowledge concepts in the learning paths recommended by FKGRec is consistent with knowledge logic, thereby supporting improved learning outcomes.

The comparative analysis underscores three primary advantages of FKGRec over baseline frameworks: (1) By integrating forgetting features with knowledge graph information, the proposed MemGNN method predicts learners' knowledge concepts at each step, enabling the simultaneous capture of both newly acquired and forgotten knowledge, overcoming the limitations of traditional approaches that focus only on new knowledge. (2) Through the incorporation of an RL algorithm, the framework recommends knowledge concepts tailored to learners' cognitive states, ensuring that the generated learning paths maintain logical sequencing between new and reviewed content, while enhancing reviewability, an aspect often overlooked by conventional methods. (3) The framework provides a clear illustration of the collaborative mechanism among its core modules, offering an intuitive explanation of why FKGRec outperforms other baseline frameworks.

6.2. Theoretical and practical implications

From a theoretical standpoint, the FKGRec framework proposed in this study contributes a novel perspective and methodological approach to learning path recommendation, particularly for dynamic path optimization in online education. First, it integrates cognitive diagnosis, knowledge concept prediction, and RL–based decision-making to construct a dynamic recommendation model that accounts for both knowledge mastery and forgetting, thereby strengthening the theoretical foundation of personalized learning path recommendations. Second, by incorporating forgetting features alongside knowledge graph information, the framework expands the scope of knowledge concept prediction and offers a more cognitively interpretable approach to modeling learning paths. Finally, by supporting the joint recommendation of new and previously learned knowledge, FKGRec fosters a theoretical shift in learning path research from focusing solely on "knowledge expansion" to emphasizing both knowledge consolidation and acquisition.

From a practical standpoint, the FKGRec framework addresses the limitations of current online education platforms, which largely prioritize recommending new knowledge. By preserving the logical sequencing of concepts, it intelligently recommends both new content and previously learned material requiring review, enabling learners to optimize outcomes within limited study time. This collaborative "learning–review" recommendation approach is particularly suited to varied scenarios, such as daily study routines and exam preparation, thereby enhancing learning efficiency and knowledge retention. Moreover, it offers a viable technical solution and practical reference for the design of innovative educational platforms, the organization of online course resources, and the delivery of personalized instructional services.

7. Conclusions

This study presents the FKGRec framework, which combines forgetting-related features with knowledge graphs to recommend learning paths that address learners' needs for both review and exploration. The framework introduces a MemGNN method to capture new and forgotten knowledge concepts during the learning process, and employs RL algorithms to recommend them in sequences consistent with the logical structure of knowledge. The experimental results confirm the effectiveness of the proposed framework.

This study also has some limitations. First, the modeling of learners' cognitive states relies only on two core features, knowledge concept mastery and learning goals, while neglecting behavioral and emotional factors that influence learning. Future work will incorporate behavioral and emotional data to enhance the modeling of learners' cognitive states, enabling a deeper understanding of their needs and supporting higher-quality learning path recommendations. Second, due to the limited availability of public datasets for generating historical learning records based on knowledge graphs, new real datasets will be developed to further verify the performance of FKGRec prior to broader application. Future research will also explore more advanced knowledge graph modeling approaches (Li et al., 2024, 2023; Shi, Li, Wang, Li, & Wu, 2025) to strengthen the model's capabilities. In addition, the reward weights of core knowledge concepts and the thresholds of forgetting factors will be adjusted to simulate different learning environments (e.g., exam preparation, daily review), thereby enhancing the model's adaptability to real-world educational scenarios.

CRediT authorship contribution statement

Yunxia Fan: Writing – review & editing, Writing – original draft, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Mingwen Tong:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition. **Duantengchuan Li:** Writing – review & editing, Writing – original draft, Project administration, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Central Special Fund for scientific research in colleges and universities (No. 30106250140), and the First-Class Disciplines Construction Project (No. 30101250301).

Data availability

Data will be made available on request.

References

Benmesbah, O., Lamia, M., & Hafidi, M. (2023). An improved constrained learning path adaptation problem based on genetic algorithm. *Interactive Learning Environments*, *31*(6), 3595–3612.

Cheng, L., Khalitov, R., Yu, T., Zhang, J., & Yang, Z. (2023). Classification of long sequential data using circular dilated convolutional neural networks. *Neurocomputing*, *518*, 50–59.

Duan, J., Zhang, P. F., Qiu, R., & Huang, Z. (2022). Long short-term enhanced memory for sequential recommendation. *World Wide Web*, *26*(2), 561–583.

Gan, W., Sun, Y., Peng, X., & Sun, Y. (2020). Modeling learner's dynamic knowledge construction procedure and cognitive item difficulty for knowledge tracing. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *50*(11), 3894–3912.

Gu, R. (2025). Personalized learning path based on graph attention mechanism deep reinforcement learning research on recommender systems. *Journal of Computational Methods in Sciences and Engineering*, 2411–2426.

Guo, K., & Zeng, G. (2023). Graph convolutional network and self-attentive for sequential recommendation. *PeerJ Computer Science*, *9*.

Huang, H., & Wang, Y. (2021). SRM: A sequential recommendation model with convolutional neural network and multiple features. In *2021 international conference on machine learning and intelligent systems engineering* (pp. 49–52).

Hwang, G. J., Kuo, F. R., Yin, P. Y., & Chuang, K. H. (2010). A Heuristic algorithm for planning personalized learning paths for context-aware ubiquitous learning. *Computers & Education*, *54*(2), 404–415.

Intayoad, W., Becker, T., & Temdee, P. (2017). Social context-aware recommendation for personalized online learning. *Wireless Personal Communications*, *97*(1), 163–179.

Kang, W. C., & McAuley, J. (2018). Self-attentive sequential recommendation. (pp. 197–206). In.

Kurilovas, E., Zilinskiene, I., & Dagiene, V. (2014). Recommending suitable learning scenarios according to learners' preferences: An improved swarm based approach. *Computers in Human Behavior*, *30*, 550–557.

Li, D., Lu, J., Wang, Z., Wang, J., Wang, X., Shi, F., et al. (2025). Recommender system based on noise enhancement and multi-view graph contrastive learning. *Applied Soft Computing*, *177*, Article 113220.

Li, D., Xia, T., Wang, J., Shi, F., Zhang, Q., Li, B., et al. (2024). SDFormer: A shallow-to-deep feature interaction for knowledge graph embedding. *Knowledge-Based Systems*, *284*, Article 111253.

Li, Z., Zhang, Q., Zhu, F., Li, D., Zheng, C., & Zhang, Y. (2023). Knowledge graph representation learning with simplifying hierarchical feature propagation. *Information Processing & Management*, *60*(4), Article 103348.

Liang, Z., Mu, L., Chen, J., & Xie, Q. (2022). Graph path fusion and reinforcement reasoning for recommendation in MOOCs. *Education and Information Technologies*, *28*(1), 525–545.

Lin, K., Li, D., Li, Y., Chen, S., Liu, Q., Gao, J., ... Gong, L. (2024). TAG: Teacher-advice mechanism with Gaussian process for reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(9), 12419–12433.

Lin, K., Li, D., Li, Y., Chen, S., & Wu, X. (2024). FHCPL: An intelligent fixed-horizon constrained policy learning system for risk-sensitive industrial scenario. *IEEE Transactions on Industrial Informatics*, *20*(4), 5794–5804.

Liu, Q., Tong, S., Liu, C., Zhao, H., Chen, E., Ma, H., et al. (2019). Exploiting cognitive structure for adaptive learning. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 627–635).

Liu, T., Wu, Q., Chang, L., & Gu, T. (2022). A review of deep learning-based recommender system in e-learning environments. *Artificial Intelligence Review*, *55*(8), 5953–5980.

Liu, H., Zheng, C., Li, D., Shen, X., Lin, K., Wang, J., ... Xiong, N. N. (2022). EDMF: Efficient deep matrix factorization with review feature learning for industrial recommender system. *IEEE Transactions on Industrial Informatics*, *18*(7), 4361–4371.

Luo, G., Gu, H., Dong, X., & Zhou, D. (2025). HA-LPR: A highly adaptive learning path recommendation. *Education and Information Technologies*.

Luo, X., Zhou, M., Li, S., & Shang, M. (2018). An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications. *IEEE Transactions on Industrial Informatics*, *14*(5), 2011–2022.

Ma, H., Huang, Z., Tang, W., & Zhang, X. (2022). Exercise recommendation based on cognitive diagnosis and neutrosophic set. In *2022 IEEE 25th international conference on computer supported cooperative work in design* (pp. 1467–1472).

Ma, Y., Wang, L., Zhang, J., Liu, F., & Jiang, Q. (2023). A personalized learning path recommendation method incorporating multi-algorithm. *Applied Sciences*, *13*(10), 5946.

Ma, D., Zhu, H., Liao, S., Chen, Y., Liu, J., Tian, F., et al. (2024). Learning path recommendation with multi-behavior user modeling and cascading deep Q networks. *Knowledge-Based Systems*, *294*, Article 111743.

Mu, M., & Yuan, M. (2024). Research on a personalized learning path recommendation system based on cognitive graph with a cognitive graph. *Interactive Learning Environments*, *32*(8), 4237–4255.

Nagatani, K., Zhang, Q., Sato, M., Chen, Y. Y., Chen, F., & Ohkuma, T. (2019). Augmenting knowledge tracing by considering forgetting behavior. (pp. 3101–3107). In.

Ren, Y., Liang, K., Shang, Y., & Zhang, Y. (2023). MulOER-SAN: 2-layer multi-objective framework for exercise recommendation with self-attention networks. *Knowledge-Based Systems*, *260*, Article 110117.

Shi, F., Li, D., Wang, X., Li, B., & Wu, X. (2025). TGformer: A graph transformer framework for knowledge graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, *37*(1), 526–541.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., et al. (2019). BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. (pp. 1441–1450). In.

Sun, Y., Zhuang, F., Zhu, H., He, Q., & Xiong, H. (2021). Cost-effective and interpretable job skill recommendation with deep reinforcement learning. In *Proceedings of the web conference 2021* (pp. 3827–3838).

Vanitha, V., & Krishnan, P. (2019). A modified ant colony algorithm for personalized learning path construction. *Journal of Intelligent & Fuzzy Systems*, *37*(5), 6785–6800.

Wang, D., Xu, D., Yu, D., & Xu, G. (2020). Time-aware sequence model for next-item recommendation. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *51*(2), 906–920.

Wang, F., Zhang, L., Chen, X., Wang, Z., & Xu, X. (2022). A personalized self-learning system based on knowledge graph and differential evolution algorithm. *Concurrency and Computation: Practice and Experience*, *34*(8), Article e6190.

Wu, Z., Li, M., Tang, Y., & Liang, Q. (2020). Exercise recommendation based on knowledge concept prediction. *Knowledge-Based Systems*, *210*, Article 106481.

Xu, W., Gao, X., Sheng, Y., & Chen, G. (2021). Recommendation system with reasoning path based on DQN and knowledge graph. In *2021 15th international conference on ubiquitous information management and communication* (pp. 1–8).

Xu, Y., Ni, Q., Liu, S., Mi, Y., Yu, Y., & Hao, Y. (2022). Learning style integrated deep reinforcement learning framework for programming problem recommendation in online judge system. *International Journal of Computational Intelligence Systems*, *15*(1).

Yuan, Y., Tang, Y., Yan, Z., Hu, M., & Du, L. (2022). KSRG:Knowledge-aware sequential recommendation with graph neural networks. In *2022 26th international conference on pattern recognition* (pp. 2408–2414).

Yun, Y., Dai, H., An, R., Zhang, Y., & Shang, X. (2024). Doubly constrained offline reinforcement learning for learning path recommendation. *Knowledge-Based Systems*, *284*, Article 111242.

Zhang, S., Hui, N., Zhai, P., Xu, J., Cao, L., & Wang, Q. (2023). A fine-grained and multi-context-aware learning path recommendation model over knowledge graphs for online learning communities. *Information Processing & Management*, *60*(5), Article 103464.

Zhang, X., Liu, S., & Wang, H. (2023). Personalized learning path recommendation for E-learning based on knowledge graph and graph convolutional network. *International Journal of Software Engineering and Knowledge Engineering*, *33*(01), 109–131.

Zhang, Q., Weng, X., Zhou, G., Zhang, Y., & Huang, J. X. (2022). ARL: An adaptive reinforcement learning framework for complex question answering over knowledge base. *Information Processing & Management*, *59*(3), Article 102933.

Zhou, Y., Huang, C., Hu, Q., Zhu, J., & Tang, Y. (2018). Personalized learning full-path recommendation model based on LSTM neural networks. *Information Sciences*, *444*, 135–152.