Knowledge-Based Systems 253 (2022) 109547

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Graph-based cognitive diagnosis for intelligent tutoring systems

Yu Su [a,b], Zeyu Cheng [c,*], Jinze Wu [d], Yanmin Dong [d], Zhenya Huang [d], Le Wu [e], Enhong Chen [d], Shijin Wang [c], Fei Xie [a]

[a] Hefei Normal University, China
[b] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, China
[c] iFLYTEK Research, China
[d] Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China, China
[e] Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology, China

ARTICLE INFO

Article history:
Received 26 January 2022
Received in revised form 22 July 2022
Accepted 22 July 2022
Available online 28 July 2022

Keywords:
Cognitive diagnosis
Graph neural networks
Interpretable machine learning

ABSTRACT

For intelligent tutoring systems, Cognitive Diagnosis (CD) is a fundamental task that aims to estimate the mastery degree of a student on each skill according to the exercise record. The CD task is considered rather challenging since we need to model inner-relations and inter-relations among students, skills, and questions to obtain more abundant information. Most existing methods attempt to solve this problem through two-way interactions between students and questions (or between students and skills), ignoring potential high-order relations among entities. Furthermore, how to construct an end-to-end framework that can model the complex interactions among different types of entities at the same time remains unexplored. Therefore, in this paper, we propose a graph-based Cognitive Diagnosis model (GCDM) that directly discovers the interactions among students, skills, and questions through a heterogeneous cognitive graph. Specifically, we design two graph-based layers: a performance-relative propagator and an attentive knowledge aggregator. The former is applied to propagate a student's cognitive state through different types of graph edges, while the latter selectively gathers messages from neighboring graph nodes. Extensive experimental results on two real-world datasets clearly show the effectiveness and extendibility of our proposed model.

1. Introduction

In recent years, online education has become increasingly popular due to the growing need for distance learning. With the help of Massive Open Online Courses (MOOCs), students from all over the world can easily access a large variety of courses. However, MOOCs usually provide students with fixed exercises and educational materials, ignoring the different levels of comprehensive ability and learning capacity among students, which leads to inflexible and inefficient applications [1–3].

The Intelligent Tutoring System (ITS) is a computer system that aims to provide different students with customized learning resources and strategies [4,5]. Therefore, estimating a student's mastery level of each skill or knowledge, called the Cognitive Diagnosis (CD) task, becomes crucial for an eligible ITS [6,7]. Fig. 1 shows an example of the CD process. Generally, the student will first take an exam to generate an exercise log that consists of questions and responses (i.e., correct or incorrect answers). Then, the CD model should infer his/her mastery degree of each skill. According to the inference result, the tutoring system will form a diagnostic report that shows customized feedback and instructions to help the student to improve mastery levels of appropriate skills. The CD process can be costly for most traditional tutoring systems relying primarily on manual rules gathered from human tutors, especially when large-scale applications are considered [8].

Solving CD problems in tutoring systems has been an active area of study. In earlier studies, researchers focus mainly on psychological algorithms since they conform to empirical rules in the educational field. Classical Test Theory (CTT) is a series of psychometric theories based on the assumption that a student's observed response is the sum of a true capability and an error term [9–11]. These methods evaluate the quality of the learning data from different aspects, including difficulty, discrimination, and reliability [12]. Specifically, difficulty reflects how difficult an item is for students; discrimination is the ability of an item to distinguish the mastery of knowledge concepts of different students; reliability reflects the consistency of all items. Item Response Model (IRM) stands for a group of psychological models that rely on the Item Response Theory (IRT) [13,14]. IRT gathers student portrait, item difficulty, and several other optional features into a logistic function to predict student performance. In real-world scenarios, IRMs are widely applied with their low complexity.

* Corresponding author.
E-mail address: zycheng2@iflytek.com (Z. Cheng).

https://doi.org/10.1016/j.knosys.2022.109547
0950-7051/

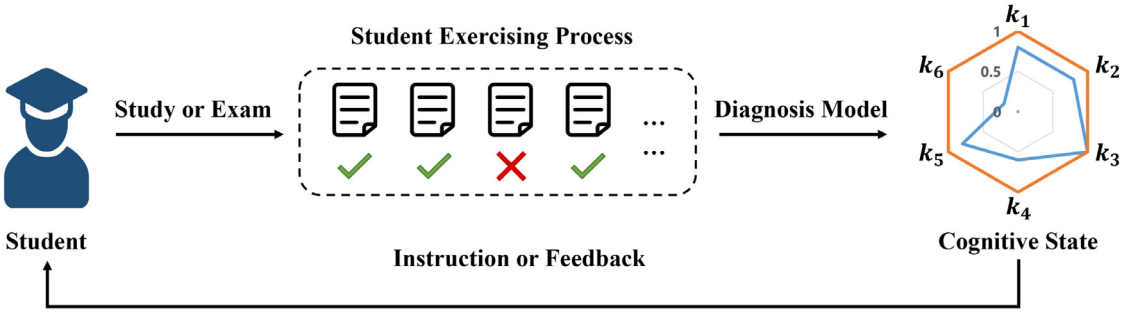


Fig. 1. An example of cognitive diagnosis.

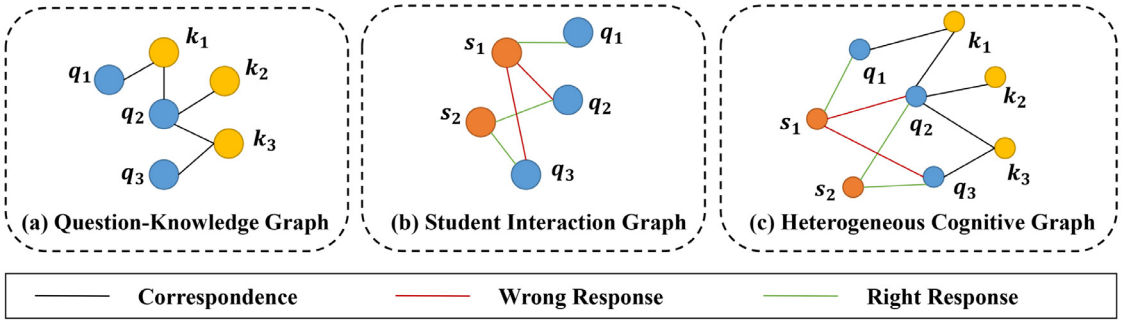


Fig. 2. The relationship among students, problems and knowledge.

Other approaches, such as DINA (deterministic inputs, noisy and gate) [15] and NIDA (noisy inputs, deterministic and gate) [16], were then proposed to bring in multiple fine-grained skill mastery. Most of these models are still based on simple interaction functions and thus show rather limited performance, especially on complex real-world data [17].

In recent years, neural networks have attracted more and more attention in many fields since deep models have shown powerful capacities to extract inner features and discover intricate feature relationships. Some attempts have also been made to employ deep models to solve CD problems. Neural Cognitive Diagnosis Model (NeuralCDM) [18] adopts a neural network to learn a more complex interaction between student factors and exercise factors. In addition, the Relation map drove Cognitive Diagnosis (RCD) [19] model incorporates structural student-exercise-concept hierarchical relations into a deep framework.

Although these models successfully bring in richer inter-relations and inner-relations between students and exercises, how information flows and propagates between different entities remains undiscovered. In a common CD scenario, the process of testing or exercising consists of two types of interactions, with which we can construct two relation graphs as shown in Fig. 2. The first one is the Question-Knowledge graph that shows the underlying skills of each question. The second graph represents the interactions between each student and all questions he/she has made.

Graph-based frameworks have become a popular research field due to their more refined structural feature extraction abilities and more explainable message merging processes. For most graph-based models, model performance is usually determined by the representations of nodes and edges. For example, Random Walk [20] generates a set of node paths by selecting random neighbored nodes recursively. These stochastic node paths are then used as the training samples to reveal co-occurrence relations among graph nodes. To improve model performance, Node2vec [21] applies a more complex sampling mechanism by combining depth-first sampling and breadth-first sampling. Graph SAmple and aggreGatE (GraphSAGE) [22] is an inductive framework that can efficiently learn embeddings for unseen nodes through a trainable function that aggregates information from neighbors. In GraphSAGE, different neighbored nodes are considered to have equivalent effects on the core node during the aggregation process, ignoring potential variance between different entities and connections. To solve this problem, Graph Attention Network (GAT) [23] introduces a self-attention mechanism to achieve automatic weight assignment for neighboring nodes.

Most of these graph-based methods are designed for isomorphic graphs where only one type of node and edge exist. Recently, heterogeneous graphs have also been studied by the community [24,25]. In a heterogeneous graph, different types of nodes and edges co-exist, making it possible to model more complex interactions in real-world scenarios. For example, in recommendation systems, graph nodes are composed of different products and users. Moreover, edges between graph nodes can have various attributes, such as clicking, purchasing, and disliking, which involves distinguishing information flows [26–28].

In tutoring systems, there are three types of entities: students, skills, and questions. In most cases, students will do exercises related to a number of skills, resulting in a ternary relation map. To bring in potential graph information, Graph-Based Knowledge Tracing (GKT) [29] applies a graph neural network to solve the knowledge tracing problem, i.e., a task of tracking a student's latent trait dynamically after each exercising action. Although GKT successfully introduces graph structure to the modeling process of student capabilities, it focuses on the pairwise relations between directly related skills. It thus ignores the potential high-level information flows through the whole graph.

To better utilize the graph topology, in this paper, we propose a novel CD framework that models student capabilities directly through a heterogeneous cognitive graph along with an innovative propagation layer and an aggregation layer. These two layers

are designed to extract high-level interactions among different entities and edges.

Our main contributions are summarized as follows:

1. To bring in richer inner- and inter-relations among students, skills, and questions, we propose a novel graph-based cognitive diagnosis framework that employs graph-level operations directly on a heterogeneous cognitive graph.
2. We design a performance-relative propagation layer that exploits exercise records to model students' learning state, along with an attentive knowledge aggregation layer that applies an attention mechanism to distinguish different types of edges.
3. Our model shows state-of-the-art performance in two real-world CD datasets. In addition, we make extra analyses that show the interpretability of the node embeddings learned by our model.

2. Related work

In this section, we will briefly introduce existing Cognitive Diagnosis frameworks and a group of graph learning methods. Firstly, we divide CD methods into two categories: the first category includes paradigms in the field of psychological measurement along with several traditional probabilistic models; the second category includes a series of deep learning models based on neural networks. Then, we will illustrate the development of the graph-based approaches.

2.1. Educational psychometric theories

Psychometric theories have been widely applied in the field of education as they are intuitive and easy to implement. Classical Test Theory (CTT) is designed to improve the reliability of the diagnosis system [9–11]. CTT makes a core assumption that each test-taker has a true score that can be regarded as a stable measurement where no error or noise occurs. Thus, the observed score of a student can be calculated by:

$$observed\_score = true\_score + error\_score, \tag{1}$$

where the error score is a standard error of measurement that is the same for all test-takers. CTT is usually used as a naive baseline since it is limited by the parameter setting where test-taker characteristics and test characteristics are bonded together.

Item Response Theory (IRT) [13,14] is a series of paradigms that have been widely applied in various domains. The core assumption IRT makes is that there exists a latent trait that represents the abilities of each test-taker. In the educational field, the latent trait can be observed through a test-taker's responses to different items. IRT employs a logistic function named Item Characteristic Curve (ICC) that fits the correlation between test-taker characteristics and item properties to predict the response (i.e., the probability of a correct answer). The following is a standard 3-parameter item response function:

$$p = c + \frac{1}{1 + e^{-D \times a(\theta - b)}}, \tag{2}$$

where θ indicates the latent ability of the test-taker; item parameters a and b represent item discrimination and item difficulty, respectively; c refers to the guessing factor that simulates the probability for a test-taker to give a correct answer by pure guessing; D denotes a constant scaling factor. In addition, a series of multidimensional IRT models (MIRT) [30,31] have been proposed to introduce multidimensional parameters, which significantly improve the modeling ability of the logistic function. To bring in more intractable and interpretable parameters, CD approaches such as Deterministic Inputs, Noisy-And gate (DINA) [15] and Noisy Inputs, Deterministic and Gate (NIDA) [16] exploit a Q-matrix that explicitly identifies specific skills of each item.

2.2. Deep-learning based models

Deep learning models have been applied in more and more fields due to their strong capability of feature extraction and feature interaction. Neural Cognitive Diagnosis (NCDM) [18] incorporates multiple neural layers to model the complex exercising process. To improve model interpretability, NCDM makes an assumption inherited from IRT paradigms that the probability of giving a correct answer increases monotonically with knowledge proficiency.

Relation Map Driven Cognitive Diagnosis (RCD) [19] models the structural relations among different entities via a multi-layer relation map. It first encodes students, exercises, and skills with trainable matrics, and then applies an attention network to perform node-level and map-level aggregation. GKT-CD [32] focuses on improving model performance by combining the CD framework and the knowledge tracing framework. It employs a gated neural network to extract students' latent traits based on the hierarchical knowledge structure.

2.3. Graph learning and graph networks

Graph Learning. Learning with graph-like data, such as social media, biological components, and financial networks, requires effective representations of the graph topology. As a fundamental sampling method, Random Walk transfers the idea of language modeling in natural language processing to the domain of node embedding. It learns node co-occurrences by generating stochastic node paths. DeepWalk [33] applies a truncated random walk algorithm to generate latent representations that contain local information. Similarly, Node2vec [34] is a framework that produces continuous representations for graph nodes by mapping each node to a low-dimensional feature space. It designs a flexible neighborhood sampling method and a biased random walk procedure that explore neighborhoods through breadth-first sampling (BFS) or depth-first sampling (DFS). For graph learning approaches, graph kernel has been an effective way of obtaining distributed graph representations, but it often requires handcrafted features. To solve this problem, Graph2vec [35] constructs rooted subgraphs to learn unsupervised and task-agnostic node embeddings.

Graph Neural Network. To perform feature extraction and other functions over the graph structure, a graph-based neural network has become a prerequisite for many tasks [36]. The Graph Neural Network (GNN) model [37] processes graph nodes by aggregating feature vectors of the neighboring nodes. After that, many variants of GNN have been proposed to achieve better performance on node and edge classification. For example, Graph Convolutional Network (GCN) [38] employs a variant of the convolutional layer that operates directly on graph structure. GCN uses an efficient layer-wise propagation rule based on a first-order approximation of spectral convolutions. As an extension of GCN, Relational Graph Convolutional Network (RGCN) [38] changes the aggregation process from local graph neighborhoods to large-scale relational nodes, dealing with highly multi-relational data characteristics. Besides, to solve unseen node problems, Graph SAmple and aggreGatE (GraphSAGE) [22] improves classical GCN in two aspects. First, it replaces whole-graph sampling with a sectional centroid node sampling, which makes large-scale distributed training and inductive learning accessible. Second, It improves model capability on neighboring aggregation through various aggregator architectures.

However, for most graph neural networks, neighboring nodes are considered equivalent even if they have different node properties and relationships with the centroid. To solve this problem, the Graph Attention Network (GAT) [23] employs a masked self-attention mechanism that allocates different weights to each

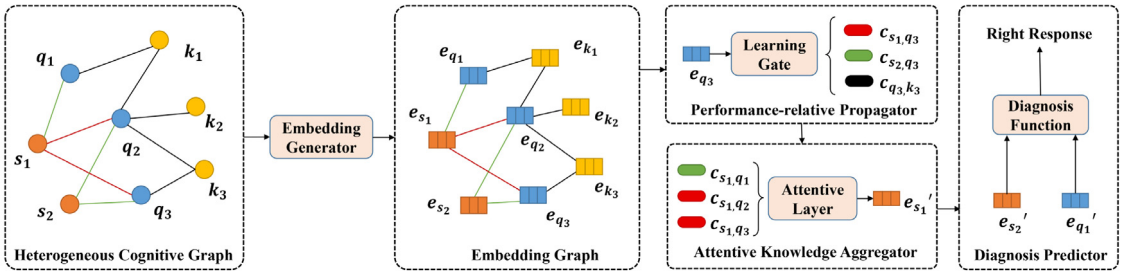


Fig. 3. The overview of the Graph-based Cognitive Diagnosis Model (GCDM).

neighboring node according to node similarities without using complex matrix operations. As GAT relies on neither the entire graph structure nor complex matrix operations to perform computation, it can be applied to address both inductive and transductive problems effectively.

Graph neural networks have been applied in many fields with their powerful processing and feature extracting capabilities on graph-like data. For example, in recommender systems, a graph structure can be formed by users, items, and item properties. Graph neural networks are then constructed on the interaction graph to extract feature representations and user portraits [39,40]. In social networks, GNNs are fundamental tools for detecting anomalies within emails and messages [41,42]. In medical treatment, researchers employ variants of GNN to perform interaction prediction among drugs and chemicals.

In the field of education, some attempts have also been made to exploit graph-based approaches in the knowledge tracing task that aims to estimate the possibility for a student to answer a specific question correctly [29,43,44]. Although graph-based knowledge tracing models show competitive capabilities, they only use the graph structure to obtain question embeddings, ignoring potential high-level interactions among different entities and edges. Moreover, as far as we know, the employment of graph neural networks in the cognitive diagnosis task is still undiscovered.

3. Problem definition

In this section, we first introduce the concept of the heterogeneous cognitive graph. Then we formally define the process of graph-based cognitive diagnosis.

3.1. Heterogeneous cognitive graph

To discover the information contained within different types of entities (i.e., students, knowledge skills, and questions) and edges, we propose a heterogeneous cognitive graph to represent the interactions among nodes in the task of cognitive diagnosis. As shown in Fig. 2, the relationships among students, knowledge skills, and questions can be represented by three forms of graphs, namely the Question-Knowledge Graph, the Student Interaction Graph, and the Heterogeneous Cognitive Graph. Specifically, the Question-Knowledge Graph reflects the inclusion relations between questions and knowledge skills. The Student Interaction Graph shows the student responses to different questions. For example, in Fig. 2, it can be seen that question q_2 includes all three knowledge skills k_1, k_2 and k_3, and student s_1 answers question q_1 correctly, while gives wrong answers to question q_2 and q_3. By combining the above two graphs, we obtain the Heterogeneous Cognitive Graph that contains three types of nodes and edges, showing the complete exercising process of all students.

The Heterogeneous Cognitive Graph can be denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{V_s, V_q, V_k\}$ is the complete set of graph nodes. The node set $\mathcal{V}$ consists of M student node indexes ($V_s = \{s_1, s_2, \ldots, s_M\}$), N question node indexes ($V_q = \{q_1, q_2, \ldots, q_N\}$), and K knowledge skill node indexes ($V_k = \{k_1, k_2, \ldots, k_K\}$). $\mathcal{E} = \{E_0, E_1, E_2\}$ denotes the complete set of graph edges. E_0 and E_1 represent the set of incorrect and correct responses between students and questions, respectively. E_2 is the set of inclusion relations between questions and knowledge skills. For example, if student s_i gives a correct answer to question q_j, the edge set E_1 will then include an edge of $e_{i,j}$; if question q_j contains knowledge skills k_{l_1} and k_{l_2}, the edge set E_2 will then include two edges e_{j,l_1} and $e_{j,l2}$. For convenience, we use a relation indicator $rel \in \{0, 1, 2\}$ to represent incorrect response, correct response, and question-skill inclusion, respectively.

3.2. Graph-based cognitive diagnosis

Given the Heterogeneous Cognitive Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the goal of the cognitive diagnosis task is to model each student's latent portrait (or cognitive state) and estimate his/her response to target questions. As the response relationships are represented by edges E_0 and E_1, the response prediction task is regarded as an edge type prediction task. Thus, considering a student node s_i and a question node q_j, the CD model aims to predict the relation type $rel_{i,j} \in \{0, 1, 2\}$.

4. Graph-based cognitive diagnosis model

In this section, we will introduce our Graph-based Cognitive Diagnosis Model (GCDM) in detail. Specifically, we will first present the overall model structure of GCDM. Then we will show the detailed design of the novel propagator and aggregator.

4.1. Overview

As shown in Fig. 3, there are four main components in GCDM: (1) The embedding generator is responsible for vectorizing students, questions, and knowledge skills to dense embeddings. (2) The performance-relative propagator is designed to infer students' cognitive states with a learning gate that gathers information from the response edges. (3) The attentive knowledge aggregator then adaptively combines cognitive states from neighboring nodes to form a more comprehensive and updated state for the core student node. (4) The diagnosis predictor uses the updated student and question embeddings to predict the student's response (i.e., the type of the relation edge). We will introduce more technical details in the following subsections.

4.2. Embedding generator

For deep learning approaches, converting input features to dense vectors is usually necessary for better modeling capabilities [45]. In the CD task, the original inputs are in the form of node

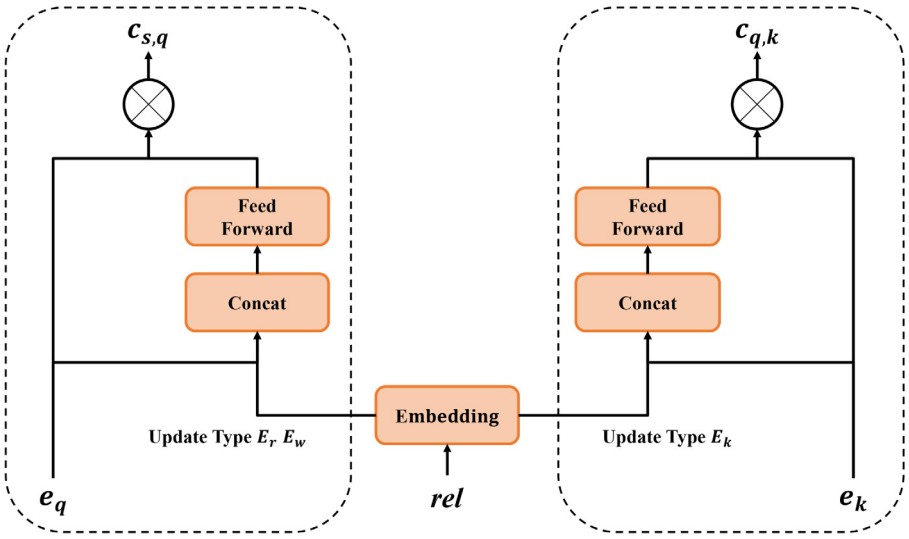


Fig. 4. Learning gate in performance-relative propagator.

indexes (categorical data), making it hard to obtain information-rich input features. One-hot encoding is a simple but commonly-used method that converts each categorical value into a new categorical column with a binary value of 1 or 0. However, it is inevitable to encounter the curse of dimensionality for datasets with a large number of distinct feature labels — various phenomena arise when a complex modeling process is performed on high-dimensional spaces with long and sparse node vectors. Therefore, we employ an embedding generator to convert the entities in the cognitive graph to dense node embeddings.

In this work, we employ a trainable matrix to learn node embeddings for different entities. Each row of the matrix is associated with a specific node index, with which we can look up for the corresponding representations for the input node. After initialization, the node embeddings are learned during the training process.

The embedding generator encodes students, questions, and knowledge skills to d-dimensional dense vectors, which be formulated as follows:

$$E = [\underbrace{\mathbf{e}_{s_1}, \ldots, \mathbf{e}_{s_N}}_{\text{students embeddings}}, \underbrace{\mathbf{e}_{q_1}, \ldots, \mathbf{e}_{q_M}}_{\text{questions embeddings}}, \underbrace{\mathbf{e}_{k_1}, \ldots, \mathbf{e}_{k_K}}_{\text{skills embeddings}}], \tag{3}$$

where $E \in R^{|\mathcal{V}| \times d}$ is a trainable embedding matrix that is initialized randomly. For any input node, we can use the node index id to find the corresponding row vector from the matrix E:

$$\mathbf{e}_{id} = E[id]. \tag{4}$$

It is noted that there are many other methods to embed the entities in the cognitive graph (e.g., NCDM [18], DINA [15]). In this paper, we choose a relatively simple approach since embedding methods or initialization algorithms are not the core contributions and usually have limited effects on the model performance.

4.3. Performance-relative propagator

In the CD task, the primary source for inferring a student's cognitive state is his/her exercising or testing performance, i.e., the responses to different questions. Intuitively, if a student gives correct answers to questions related to a particular knowledge concept, we believe the student has a strong mastery of this knowledge concept. On the contrary, if a high proportion of incorrect answers is observed, we believe that the student has a relatively weak mastery. To precisely model students' cognitive states, we design a performance-relative propagator that learns the information flow through edges between student and question nodes.

The cognitive state ($\mathbf{c}_{\mathbf{s,q}}$) reflected in an interaction between student s and question q is mainly determined by two factors. One is the knowledge state revealed in question q ($\mathbf{e}_{\mathbf{q}}$). The other is the cognitive degree that the student has achieved on question q, denoted by $\mathbf{l}_{\mathbf{s,q}}$. To obtain the cognitive degree $\mathbf{l}_{\mathbf{s,q}}$, we propose a learning gate that exploits the question features and the student performance at the same time. As shown in Fig. 4, the learning gate uses question embedding $\mathbf{e}_{\mathbf{q}}$ and the edge type rel as the inputs. Then, a neural network is applied to merge the inputs and form the cognitive degree $\mathbf{l}_{\mathbf{s,q}}$. Specifically, if student s answers question q incorrectly, we assume that the interaction makes no contribution to the mastery level. Therefore, the cognitive degree $\mathbf{l}_{\mathbf{s,q}}$ is calculated by:

$$\mathbf{l}_{\mathbf{s,q}} = \mathbf{W}_{\mathbf{l,rel}} \cdot [\mathbf{e}_{\mathbf{q}}, \mathbf{W}_{\mathbf{rel}} \cdot rel] + \mathbf{b}_{\mathbf{l,rel}}, \ rel \in \{0, 1\}, \tag{5}$$

where $\mathbf{W}_{\mathbf{l,rel}} \in R^{d \times 2d}$ and $\mathbf{b}_{\mathbf{l,rel}} \in R^{d \times 1}$ are trainable model weights of the learning gate. $W_{rel} \in R^{d \times 1}$ is a trainable matrix that controls the final cognitive degree according to the student response. It can be seen that if $rel = 0$, W_{rel} produces no gain to $\mathbf{l}_{\mathbf{s,q}}$.

To obtain the cognitive state $\mathbf{c}_{\mathbf{s,q}}$ from a student–question interaction, we multiply the knowledge state and the cognitive degree element-by-element as follows:

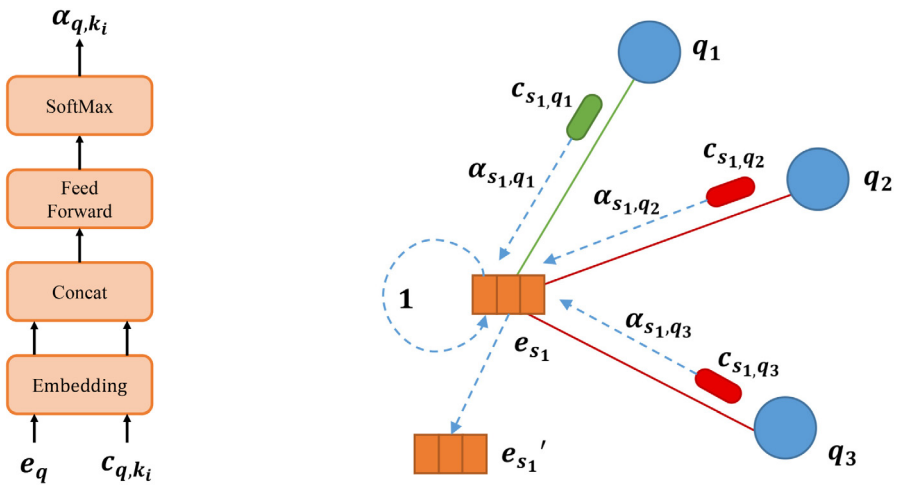
$$\mathbf{c}_{\mathbf{s,q}} = \mathbf{e}_{\mathbf{q}} \odot \mathbf{l}_{\mathbf{s,q}}. \tag{6}$$

In the Heterogeneous Cognitive Graph, each question can be related to more than one knowledge skill. Therefore, we use the knowledge inclusion state $\mathbf{c}_{\mathbf{q,k}}$ to distinguish questions by their neighboring skills. Similarly, the inclusion state for skill k in question q is determined by two factors: one is the knowledge skill embedding $\mathbf{e}_{\mathbf{k}}$; the other is the inclusion degree $\mathbf{l}_{\mathbf{q,k}}$ indicating the level for the knowledge contained in the question. The inclusion state is then calculated by:

$$\mathbf{l}_{\mathbf{q,k}} = \mathbf{W}_{\mathbf{l,rel}} \cdot [\mathbf{e}_{\mathbf{k}}, \mathbf{W}_{\mathbf{rel}}] + \mathbf{b}_{\mathbf{l,rel}}, \ rel = 2, \tag{7}$$

$$\mathbf{c}_{\mathbf{q,k}} = \mathbf{e}_{\mathbf{k}} \odot \mathbf{l}_{\mathbf{q,k}}. \tag{8}$$

The cognitive state and the inclusion state will be propagated along the graph edges and then aggregated to update the target student node.



(a) Attention Layer (b) Aggregator

Fig. 5. Attentive knowledge aggregator.

4.4. Attentive knowledge aggregator

In practice, students' mastery of a knowledge concept can be more accurately inferred by repeated tests on the same concept. Similarly, we can mine the students' cognitive states by integrating a large number of students' answers from practice logs. It is also reflected in the cognitive graph. The student and the practiced questions will form a subgraph, which shows the student's specific cognitive structure. In the subgraph, the student node can aggregate his/her cognitive states from the neighbors based on the interaction between the student and questions.

However, different questions have different investigative abilities and test students from different perspectives [46,47]. For example, some questions may be too simple to distinguish students' cognitive states. Therefore, the test results of these questions are not referential enough. In addition, some questions investigate two knowledge concepts, "Division" and "Equation", while some of these questions focus on the knowledge concept "Division" and others focus on the "Equation". Therefore, the reflexes of different response relationships between students and questions on students' cognitive states are not the same. Instead of considering different neighbors equivalent to the core node, we propose the attentive knowledge aggregator to distinguish the effect of different cognitive interaction processes on students' cognitive states, as shown in Fig. 5. In particular, the attentive knowledge aggregator adaptively aggregates different cognitive states from neighbor nodes to obtain a comprehensive target nodes representation. The cognitive states that reflect the students' cognitive state more effectively are given a higher weight, while the cognitive states generated from the cognitive interaction process with less information may be ignored. Finally, by aggregating the reflex from different cognitive interactions, we can obtain the comprehensive cognitive states of the student nodes.

Specifically, for student node s, the attentive knowledge aggregator first calculates the coefficient of importance on different cognitive interactions in the attentive layer. For each cognitive interaction between the student node and the neighbor question node q_i in $\mathcal{N}$, we obtain the coefficient of importance a_{s,q_i} with the cognitive state c_{s,q_i} and the student node embeddings e_s, which is denoted as:

$$a_{s,q_i} = W_a \cdot [W_n \cdot \mathbf{e_s}, W_n \cdot \mathbf{c_{s,q_i}}], \ q_i \in \mathcal{N}. \tag{9}$$

It is worth noting that $W_n \in R^{hd \times d}$ is a linear mapping matrix with shared parameters, which can transfer the node features. In particular, there is a multi-head attentive layer when $h > 1$. Besides, $W_a \in R^{d \times 1}$ maps the spliced higher-dimensional features to a real number a_{s,q_i}. Then we normalize the coefficients of importance a_{s,q_i} from all the interactions to obtain the attention weight as:

$$\alpha_{s,q_i} = \frac{\exp\left(LeakyReLU(a_{s,q_i})\right)}{\sum_{q_j \in \mathcal{N}} \exp\left(LeakyReLU(a_{s,q_j})\right)}, \tag{10}$$

that is, the weight of the corresponding cognitive states. Finally, we weighted aggregate the cognitive states of all neighbor nodes and added them with the original representation of the student node $\mathbf{e_s}$ to obtain the new representation of the cognitive states:

$$\mathbf{e_s'} = \mathbf{e_s} + \sum_{q_i \in \mathcal{N}} \alpha_{s,q_i} \times \mathbf{c_{s,q_i}}. \tag{11}$$

Similarly, the importance of different knowledge concepts also varies for different questions. For a question that tests multiple knowledge concepts, some knowledge concepts are the core test sites of this question, but other knowledge concepts are often secondary test sites. Therefore, we use the same layer to calculate the attention weights of the cognitive states of different knowledge concept nodes and aggregate them into the question node representation as:

$$a_{q,k_i} = W_a \cdot [W_n \cdot \mathbf{e_q}, W_n \cdot \mathbf{c_{q,k_i}}], k_i \in \mathcal{N}, \tag{12}$$

$$\alpha_{q,k_i} = \frac{\exp\left(LeakyReLU(a_{q,k_i})\right)}{\sum_{k_j \in \mathcal{N}} \exp\left(LeakyReLU(a_{q,k_j})\right)}, \tag{13}$$

$$\mathbf{e_q'} = \mathbf{e_q} + \sum_{k_i \in \mathcal{N}} \alpha_{q,k_i} \times \mathbf{c_{q,k_i}}. \tag{14}$$

4.5. Diagnosis predictor

With the Propagation and Aggregation on the heterogeneous cognitive graph, we update the embeddings of student nodes and questions nodes. We finally predict the student performance in the diagnosis predictor.

In our paper, considering the linking between student nodes and question nodes, we define the student performance prediction in the form of an edge prediction task on the cognitive graph. For certain an edge between the student node s_i and question node q_j, we expect to predict the type of the edge, which represents whether the student answers the question correctly.

Therefore, with the student–question node pair (s_i, q_j), we calculate the performance of the student s_i on the question q_j as:

$$p = \mathcal{F}(\mathbf{e}'_{\mathbf{s_i}} + \mathbf{e}'_{\mathbf{q_j}}). \tag{15}$$

Similarly, taking the correlation between student nodes and knowledge concept nodes into count, we can define the cognitive states on knowledge concepts from the cognitive graph as:

$$h = \mathcal{F}(\mathbf{e}'_{\mathbf{s_i}} + \mathbf{e}_{\mathbf{k}}). \tag{16}$$

It is noting that the $\mathcal{F}(\cdot)$ is a diagnosis function, which uses both the student and question node representations to obtain the final prediction. In our paper, we simply implement it with a linear function to verify the effectiveness of our graph-based cognitive diagnosis model, which can be denoted as:

$$p = W_p \cdot (\mathbf{e}_{\mathbf{s_i}} + \mathbf{e}_{\mathbf{q_j}}) + b_p. \tag{17}$$

We use the cross entropy loss as the loss function when optimizing our model. We minimize the gaps between the prediction p and the ground truth result y as:

$$loss = \sum_i (y_i \log p_i + (1 - y_i) \log(1 - p_i)). \tag{18}$$

Finally, our model has strong extendibility. Apparently, with the representations of students and questions, the existing cognitive diagnostic methods can be attached following our model. In that case, the GCDM is regarded as an encoding layer, and we can replace the student and question representations in other cognitive diagnosis models with the $\mathbf{e_s}$ and $\mathbf{e_q}$ in GCDM. Take some important cognitive diagnosis models for example as follows:

IRT. IRT is a cognitive diagnosis method of modeling students' cognitive states and questions' parameters by a logistic-like function. Take the typical 2-parameter IRT function for example:

$$p = \frac{1}{1 + \exp(-D \times a \times (\theta - b))}, \tag{19}$$

where θ is the student ability, b is the question difficulty and a is the question discrimination. D denotes a constant scaling factor. To extend from IRT, we replace the student representation and question difficulty with unidimensional e_s and e_q. And a can be defined as a trainable parameters. The IRT-based extension can be denoted as:

$$p = \frac{1}{1 + \exp(-D \times a \times (\mathbf{e_{s_i}} - \mathbf{e_{q_j}}))}. \tag{20}$$

MIRT. MIRT is a multidimensional extension of the IRT, which models students and exercises on the multiple knowledge perspective, which is denoted as:

$$p = \frac{1}{1 + \exp(-\mathbf{Q_e} \times (\mathbf{h_s} - \mathbf{h_e}))}, \tag{21}$$

where $\mathbf{h_s}$ is the student representation, the $\mathbf{h_e}$ is the question representation and Q_e is the Q-matrix, which is a multi-hot vector and represents the knowledge concepts related to the question.

To extend from MIRT, we replace the student representation and question representation with e_s and e_q. The MIRT-based extension can be denoted as:

$$p = \frac{1}{1 + \exp(-\mathbf{Q_e} \times (\mathbf{e_{s_i}} - \mathbf{e_{q_j}}))}. \tag{22}$$

NCDM. NCDM encodes students and questions and automatically fits the complex interaction functions between students and questions based on the networks, which is defined as

$$p = F(\mathbf{Q_e} \cdot (\mathbf{h^s} - \mathbf{h^{diff}}) * h^{disc}), \tag{23}$$

where $\mathbf{h^s}$ is the student representation, $\mathbf{h^{diff}}$ is the question difficulty and h^{disc} is the question discrimination, respectively, and $*$ denotes scalar multiplication. Besides, Q_e is the Q-matrix, which is a multi-hot vector and represents the knowledge concepts related to the question. $F(\cdot)$ is a multi-layer full connection layer that fits the complex interaction functions between students and questions.

To extend from NCDM, we regard our GCDM as a pre-trained encoder and similar to IRT and MIRT, we replace the student representation and question difficulty with e_s and e_q. And $\mathbf{h^{disc}}$ can be defined as a trainable parameters. The NCDM-based extension can be denoted as:

$$p = F(\mathbf{Q_e} \odot (\mathbf{e_{s_i}} - \mathbf{e_{q_j}}) * h^{disc}). \tag{24}$$

5. Experiments

In this section, we conduct extensive experiments to demonstrate the effectiveness of our proposed GCDM. Specifically, we first describe the real-world datasets used in our experiments (Section 5.1). Then we introduce the experimental setups, including data partitioning, model implementations, and the compared models (Section 5.2). After that, we carry out experiments from the following three aspects (Section 5.3): (1) We demonstrate the performances of graph-based cognitive diagnosis on the heterogeneous cognitive graph; (2) We evaluate the performance of GCDM in the cold start situation from exercise perspective; (3) We analyze the correlation of graph nodes. (4) We calculate the consistency of students' cognitive states and performances, and then conduct the case study to verify the interpretation of our model.

5.1. Datasets

In the experiments, we used two real-world educational datasets which are commonly used for student performance prediction, namely ASSIST0910 (ASSISTments 2009–2010 "skill builder"[1]) and ASSIST2017 (ASSISTments 2017[2]). Both the open datasets are collected from the ASSISTments online tutoring systems, which record the mathematics logs of students. To mine the high-order relationship in the cognitive process of students, we build the heterogeneous cognitive graph based on the datasets.

ASSIST0910. Based on ASSIST0910, we build a cognitive graph with 21,998 nodes which consist of 4128 student nodes, 17,746 question nodes and 123 knowledge concepts nodes (i.e., "Equation" and "Conversion"), and 581,744 edges (with reverse edges) from logs which consist of 3 types (right answer, wrong answer, and question-knowledge correspondence).

ASSIST2017. Based on ASSIST2017, we build a cognitive graph with 4974 nodes which consist of 1709 students nodes, 3162 question nodes and 102 knowledge concepts nodes, and 1,891,956 edges which consist of the same 3 types as the former graph.

Furthermore, Table 1 shows more detailed statistics of the two datasets.

5.2. Experimental setups

5.2.1. Data partitioning

On both datasets, we sample 80% of edges between students and questions for each student, along with all edges between questions and knowledge concepts, and then use them as the training set. The remaining 20% of edges between students and

[1] https://github.com/bigdata-ustc/EduData.

[2] https://github.com/bigdata-ustc/EduData.

Table 1
Statistics of the datasets: ASSIST0910 and ASSIST2017.

Statistics	ASSIST0910	ASSIST2017
# of nodes	21,998	4974
# of students	4128	1709
# of questions	17,746	3162
# of knowledge concepts	123	102
# of edges	581,744	1,891,956
# of records	267,415	942,816
# records per student	64.78	551.67
# knowledge concepts per item	1.37	1.0
# density	0.0012	0.0764

questions are used as the testing set for the corresponding student. We also sample 10% from the training set to form the validation set to develop our models. For each model we evaluate, we run 5 times on each dataset and use the average result as the final result. Especially, to avoid the leakage of validation set and test set data in the prediction process along with graph structure whether it is training or testing, we mask all edges in the validation set and test set and build the heterogeneous cognitive graph only with the training set nodes and edges. In this way, the model only can use the training data when sampling the neighbor nodes and edges.

5.2.2. Training settings

We initialize all parameters in GCDM with Xavier initialization following [48] with the uniform distribution in the range $(-\sqrt{(6/n_i + n_o)}, \sqrt{(6/n_i + n_o)})$, where n_i and n_o are the dimensions of the input and output, respectively. We then train the GCDM with mini-batches of 1024 and a learning rate of 0.0005 under the Adam optimizer. We also implement algorithms including Dropout [49] (dropout rate $= 0.4$) and gradient clip (clip value $= 5$) to avoid over-fitting, which generates a slight improvement in model performance and thus will not be discussed in detail. On both datasets, we train our models for at most 50 epochs to obtain the best performances.

5.2.3. Model comparison

To illustrate the effectiveness of our proposals, we implement several existing cognitive baselines[3] which are trained on the record logs:

1. IRT [50]: IRT is a cognitive diagnosis method of modeling students' cognitive states and questions' parameters by a logistic-like function.
2. MIRT [50]: MIRT is a multidimensional extension of the IRT, which models students and exercises on the multiple knowledge perspectives.
3. DINA [15]: DINA is a cognitive diagnosis method of modeling each student's knowledge proficiency by a binary vector with Q-matrix.
4. NCDM [18]: NCDM is one of the most recent CD models based on neural networks. It encodes the student and questions and automatically fits the complex interaction functions between students and questions based on the networks.
5. RCD [19]: RCD focuses on the multiple relations and models the interactive and structural relations via a multi-layer student-exercise-concept relation map.

Besides, we also compare with some graph neural networks which are trained on the cognitive graph to verify our methods:

1. GCN [51]: GCN proposes a graph convolution method that updates the node embedding with all the neighbor nodes.
2. RGCN [38]: RGCN focuses on a different type of edges in the graph and it updates the node embeddings from neighbor nodes by types.
3. SAGE [22]: SAGE samples part of the neighbor nodes and aggregates the nodes to the source node to update the new embedding.
4. GAT [23]: GAT learns the importance of neighbor nodes and employs a weighted sum function to update the source node embedding.
5. GIN [37]: GIN introduces the MLP (Multilayer Perceptron) to learn the aggregation function to aggregate the node embedding.

In addition, to understand the specific effects brought by each key proposal in GCDM, we compare the complete GCDM with two simplified variants:

1. w/o propagator: we implement a simplified model based on GCDM without a performance-relative propagator. This model ignores the impact on the cognitive states of students from response relationships. Instead, in this model, source node delivers the node embedding to the neighbor nodes along edges directly, that is, $\mathbf{c}_{\mathbf{s},\mathbf{q}} = \mathbf{e}_{\mathbf{q}}$ transformed from Eq. (6).
2. w/o aggregator: we implement a simplified model based on GCDM without attentive knowledge aggregator. This model ignores differences among questions and differences among knowledge concepts. Instead, the model allows nodes equally aggregate all neighbor nodes, that is, $\mathbf{e}'_{\mathbf{s}} = \sum_{q_i \in \mathcal{N}} \mathbf{c}_{\mathbf{s},\mathbf{q_i}}/N$ transformed from Eq. (11).

In the following experiments, all the above-mentioned baselines and our proposed models are implemented by PyTorch. For fairness, all the methods are trained with the optimal settings described in their original paper to guarantee the performances. All models are trained on the same Linux server with four 2.30 GHz Intel Xeon E5-2650 CPUs, two NVIDIA Tesla M40 GPUs, and 256 GB memory to achieve the best performance for comparison.

5.3. Results and analysis

5.3.1. Evaluation metrics

In our experiments, we process cognitive diagnosis with the student performance prediction task. To verify the prediction, we employ some widely used metrics [52,53]. To be specific, we use the ROC Curve (AUC) and Prediction Accuracy (ACC) to measure the prediction performance from a classification perspective in the range of $[0, 1]$. The larger the values are, the better the results. Besides, we use the Root Mean Square Error (RMSE) to quantify the gaps between predictions and true responses. The lower RMSE is, the better the model performs.

5.3.2. Performance prediction

To verify the effectiveness of our proposed models, we first evaluate the accuracy performances on student performance prediction tasks. In the experiment, we selected all the baselines mentioned in Section 5.2.3, including CD baselines and GNN baselines for comparison. We use the metrics of ACC, AUC, RMSE mentioned in our experiments. Table 2 lists the overall results on both datasets with the evaluation metrics mentioned.

There are some key observations: (1) GCDM performs better than existing cognitive diagnosis methods. It shows our graph-based cognitive diagnosis model on the heterogeneous cognitive graph is more effective, since GCDM models the high order relationships from the graph. (2) GCDM also outperforms the general

[3] https://github.com/bigdata-ustc/EduCDM.

Table 2
Student prediction performances for metrics on both datasets.

Methods	ASSIST0910			ASSIST2017		
	ACC	RMSE	AUC	ACC	RMSE	AUC
IRT	0.654	0.472	0.681	0.658	0.464	0.668
MIRT	0.707	0.461	0.716	0.668	0.461	0.678
DINA	0.644	0.495	0.680	0.613	0.519	0.654
NCDM	0.726	0.441	0.752	0.685	0.453	0.699
RCD	0.724	0.427	0.761	0.694	0.450	0.709
GCN	0.710	0.461	0.725	0.652	0.467	0.668
RGCN	0.718	0.449	0.749	0.682	0.455	0.697
SAGE	0.723	0.437	0.751	0.690	0.452	0.704
GIN	0.721	0.437	0.750	0.689	0.452	0.701
GAT	0.721	0.442	0.751	0.676	0.464	0.681
GCD-IRT	0.669	0.451	0.720	0.673	0.457	0.687
GCD-MIRT	0.719	0.454	0.738	0.692	0.450	0.705
GCD-NCDM	0.722	0.439	0.747	0.684	0.456	0.693
w/o aggregator	0.726	0.428	0.763	0.692	0.450	0.707
w/o propagator	0.725	0.427	0.761	0.691	0.451	0.709
GCDM	**0.729**	**0.425**	**0.766**	**0.695**	**0.448**	**0.712**

Table 3
Student prediction performances of cold start new questions for metrics on both datasets.

Methods	ASSIST0910			ASSIST2017		
	ACC	RMSE	AUC	ACC	RMSE	AUC
RCD	0.656	0.472	0.604	0.583	0.491	0.579
GCN	0.641	0.481	0.599	0.578	0.498	0.529
RGCN	0.655	0.489	0.572	0.581	0.491	0.559
SAGE	0.527	0.611	0.515	0.542	0.521	0.510
GIN	0.621	0.500	0.586	0.569	0.502	0.516
GAT	0.652	0.535	0.611	0.525	0.502	0.526
GCDM	**0.664**	**0.469**	**0.624**	**0.591**	**0.487**	**0.580**

graph neural networks. It shows our method that takes characters in the cognitive graph into account can mine not only the topology information of the graph but also the cognitive relationship. It will benefit the cognitive diagnosis issue. (3) Compared with the simplified models (w/o aggregator and w/o propagator), the complete GCDM also has better performance. It shows both the performance-relative propagator and attentive knowledge aggregator play a role in cognitive diagnosis on the heterogeneous cognitive graph. We should fully explore the topological information and cognitive information from each student's cognitive subgraph. (4) GNN-based models perform relatively well. It indicates that the cognitive graph can provide more cognitive information from the interactions among students, questions, and knowledge tracing, than only considering the pair-wised interactions. (5) The extended models, GCD-*, get relatively good performances. It shows that the information from the cognitive graph plays an important role in cognitive state modeling. Based on the cognitive graph, we can obtain more comprehensive representations of the cognitive states of students and questions. More efficient inputs lead to more accurate outputs in the cognitive diagnosis models.

5.3.3. Cold start evaluation

Compared with non-graph-based cognitive diagnosis methods, graph-based methods can model the high order relationships of questions based on the knowledge concept nodes, where the knowledge system is usually public and complete. Therefore, graph-based methods may be more robust when meeting new questions [54]. The cognitive diagnosis methods only based on response logs cannot work on new questions [55], since the questions are encoded into initialized embedding. To verify the effectiveness of our proposed models in the cold start situation, we re-partition the datasets. We use a similar method to partition datasets as described in the normal experiment setting. For each student node, we sample 80% of edges between students and questions, along with all edges between questions and knowledge concepts, and then use them as the training set. The remaining 20% of edges between students and questions are used as the testing set for the corresponding student. The only difference is that the questions in the test set are selected to make sure they have not been seen during the training process. In other words, the model is trained on edges from known question nodes, but it is tested on edges related to unseen question nodes. We select all the graph-based baselines, including RCD and GNN baselines for comparison. Similarly, we use the metrics of ACC, AUC, and RMSE mentioned in our experiments. Table 3 lists the results on both datasets on the graph-based methods mentioned.

We first calculate the average scoring rate of the test set in a cold start situation over the two datasets. The scoring rate is 0.652 in ASSIST0910 and 0.422 in ASSIST2017, which can be regarded as the expected accuracy with a complete guess process on the cold question. From Table 3, we can draw the following conclusions: (1) Most graph-based methods perform better than the expected indicators. It shows that the graph-based methods can work on the cold questions. This is probably because graph-based methods can obtain the relatively appropriate representation of cold questions by the question-knowledge edges from the cognitive graph. When it comes to getting a representation of a cold question, traditional methods usually only provide a randomly initialized representation, resulting in unavailability. The graph-based method will generate it based on the graph structure where the question is located in. According to the subgraph of the cold question, the information from related entities nodes, such as related knowledge concepts of the question will be synthesized to obtain a more informative, more accurate representation. (2) We find that the GCDM is more robust than other basic methods. It means that GCDM can mine the high order relations between known questions and cold questions better so that it maintains relatively good performances in widely cold situations. This is because GCDM can better propagate and aggregate the information of nodes to obtain a reasonable representation of the central node.

5.3.4. Node correlation analysis

As mentioned before, we build a heterogeneous cognitive graph based on the interaction data between students and questions. We then define the student performance prediction as an edge prediction task for cognitive diagnosis. In this case, we conduct some interesting visualization experiments on the heterogeneous cognitive graph. Therefore we can analyze the entities in the cognitive process, such as student, question, and knowledge concept from a graph perspective. Specially, we illustrate student nodes, question nodes, and knowledge concept nodes, respectively.

Student node distribution. In this experiment, we visualize the student node embedding e_s from GCDM by reducing their dimension with t-SNE [56], which is commonly used to reduce the dimension of vector to 2D data. Then we color the student nodes by the corresponding student's average response scores in Fig. 6. Especially, in this figure, we color the students who perform better lighter, while color the students who perform worse are darker to distinguish the students.

Fig. 6 shows the distribution of student nodes. It is noted that the distribution of student nodes has a close relationship with the student's average score. As we position the student nodes referring to the node embeddings, that is the student cognitive states, the student who has similar scores tend to be more contiguous. For example, the students located on the right part are mostly with better scores (more than 0.8), while the students

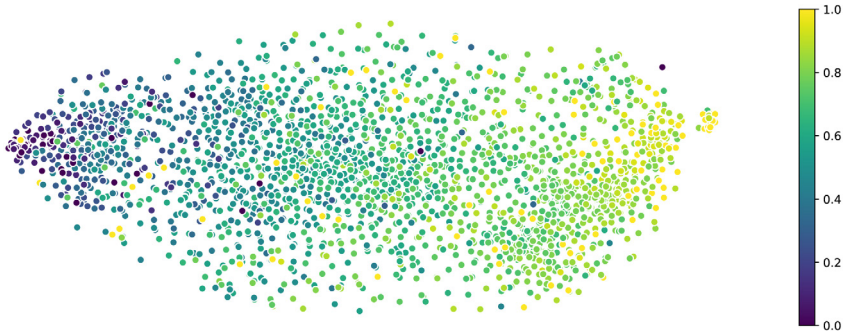


Fig. 6. The student nodes embeddings of GCDM reduced dimension by t-SNE colored with average score in ASSIST0910.

located on the left part are mostly with worse scores (less than 0.2). Generally, similar cognitive subgraphs of student nodes lead to similar interactions, which further leads to similar student embeddings and overall grades. It also proves our model can identify and construct higher-order connections among students who perform similarly from the heterogeneous cognitive graph. In particular, GCDM can define similar student nodes based on the neighbor subgraph structure.

Question node clustering. In this experiment, we expect to validate whether graph-based methods can mine the high order relationships among question nodes. Specifically, we choose graph-based methods including our GCDM to analyze the node embedding on clustering impressions, since in these methods, we do not initialize the question nodes with knowledge concept information, nor do we explicitly update some elements of the cognitive states by introducing the Q-matrix. Specifically, we visualize the node embedding by reducing their dimension with t-SNE [56]. For better illustration, we choose questions from four different knowledge concepts (“Equation Solving More Than Two Steps”, “Pythagorean Theorem”, “Probability of Two Distinct Events”, and “Solving for a Variable”). We then label question nodes of each knowledge concept with different colors.

Fig. 7 illustrates the distribution of question nodes’ embedding. It can be seen that questions nodes associated with the same knowledge concept tend to be located together. The distributions at the knowledge perspective in question nodes are generated entirely from the structure of the cognitive graph. It shows that the graph-based methods indeed find the high order relationship of questions from the graph structure, even though there are no direct edges among questions in the heterogeneous cognitive graph. Specifically, graph-based methods can capture the rich information from topological structures between question nodes and knowledge concept nodes and abstract the unique distribution of the question in the knowledge concepts.

We also perform a quantitative evaluation to compare the question node clustering effect of the graph-based models in our experiments. Specifically, three commonly-used validation metrics, cohesion, separation, and Calinski–Harabasz coefficient (CH), are measured on ASSIST0910. Cohesion shows the internal dispersion of clusters. The lower the cohesion, the more compact each cluster is. Separation shows the dispersion between clusters. The higher the separation, the better capability the model has in differentiating a cluster from others. CH considers both the internal dispersion of clusters and the dispersion between clusters, as it is the normalized ratio of separation to cohesion. Therefore, a higher value of CH indicates that the clusters are dense and well separated. These metrics are formulated as follows:

$$cohesion = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{m_i}\sum_{j=1}^{m_i}\|cluster_{ij} - mean_i\|\right), \tag{25}$$

$$separation = \frac{1}{n}\sum_{i=1}^{n}\|mean_i - mean\|, \tag{26}$$

$$CH = \frac{(M-n)\sum_{i=1}^{n} m_i\|mean_i - mean\|^2}{(n-1)\sum_{i=1}^{n}\sum_{j=1}^{m_i}\|cluster_{ij} - mean_i\|^2}, \tag{27}$$

where M denotes the total number of question nodes, m_i denotes the number of question nodes in cluster i, and n is the number of clusters. $cluster_{ij}$ denote the position of node j in cluster i. $mean_i$ and $mean$ are the centroid of cluster i and the centroid of all nodes, respectively.

As shown in Fig. 8, GCDM has the lowest cohesion and a relatively high CH (second only to GAT), indicating that our model generates a well-dispersed question node distribution, and the topological relationships learned from the heterogeneous cognitive graph help to mine higher-order relationships of questions. It is noted that GAT shows the highest CH as it has a lower cohesion compared to our model. This is primarily due to the direct propagation method applied in GAT that usually generates more compact clusters of connected graph nodes. Meanwhile, in GCDM, node information is propagated in a more complex and differentiated way with the performance-relative propagator, which may result in relatively sparse clusters compared to GAT. Nevertheless, GCDM still shows competitive results and overperforms most graph-based models in CH.

Knowledge concept node correlation. Then in this experiment, we expect to verify whether the relation of knowledge concept nodes can be mined from GCDM in an unsupervised way since there are no edges linking knowledge concepts in our datasets. We also visualize the knowledge concept nodes from our GCDM following the same process by reducing their dimension with t-SNE as shown in Fig. 9. Fig. 9 illustrates the distribution of knowledge concept nodes’ embedding. It can be found that the embeddings of knowledge concepts are not uniformly distributed. Instead, some nodes are clustered closer. For example, we can find the node k_1 representing the knowledge concept “Addition and Subtraction Integers” and k_2 representing the knowledge concept “Addition and Subtraction Fractions” are located nearby. Similarly, the node k_3 which represents knowledge concept “Reflection” and k_4 which represents knowledge concept “Translations” are located nearby. The “Addition and Subtraction Integers” and “Addition and Subtraction Fractions” are concepts that are very relevant in terms of knowledge. It shows that to some extent our GCDM can find the relationship of knowledge concepts based on the heterogeneous cognitive graph, even though there are no direct edges among knowledge nodes. It may lead to some unsupervised knowledge topology discovery. For example, we can annotate knowledge concepts, divide knowledge concepts and automatically discover knowledge relationships. This is because relevant knowledge concepts often appear in certain questions in form of co-occurrence, which will form the unique topology in the cognitive graph and be discovered by our method.

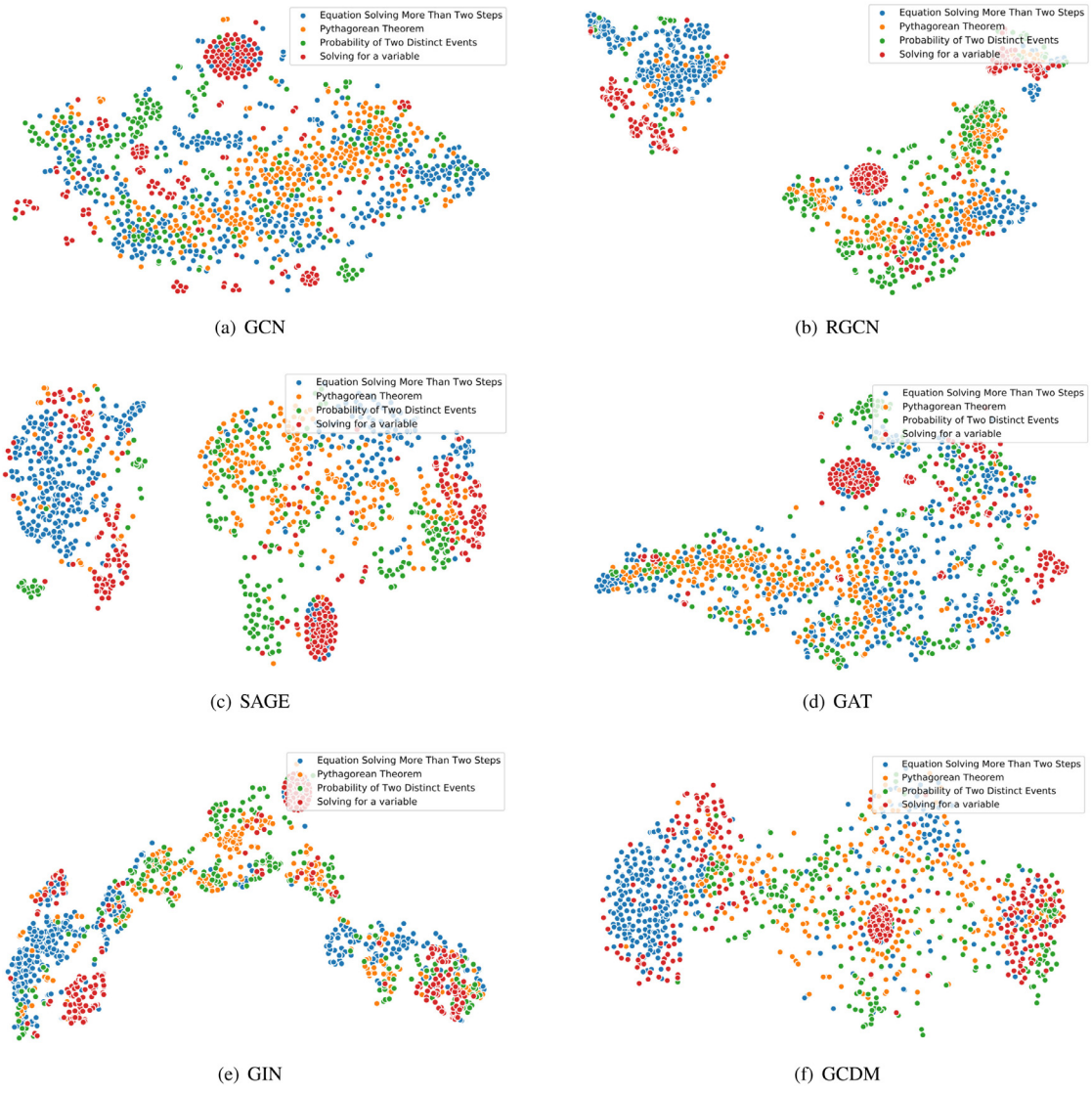


Fig. 7. The question nodes embeddings of graph-based methods reduced dimension by t-SNE on five knowledge concepts in ASSIST0910.

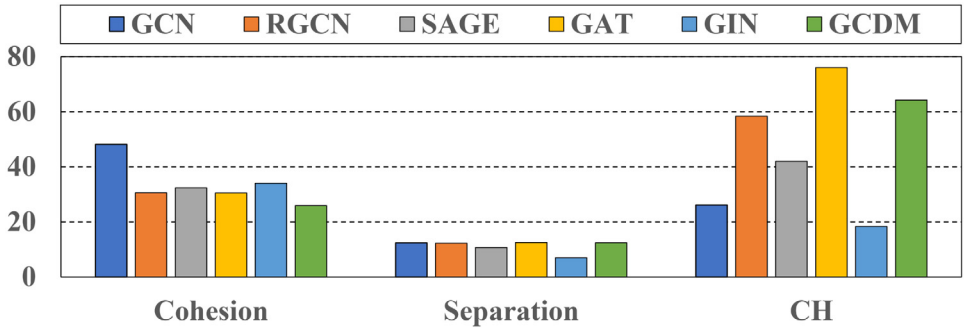


Fig. 8. Three metrics showing the question node clustering effect of each graph-based model in the experiment on ASSIST0910.

5.3.5. Model interpretation

Quantization of Interpretation. In educational scenarios, the comparability of students is also important. We need to make sure the consistency of students' cognitive states and performances to some extent. Intuitively, if student a performs better on questions on concept k than student b, he may have a better mastery of the concept. However, it is difficult to directly evaluate this part since there is no direct way to get the actual cognitive states of students. In that case, following [18,57,58], we adopt the Degree of Agreement (DOA) metric to evaluate the ranking performance of each model. Particularly, DOA result on a specific knowledge concept k is defined as:

$$DOA(k) = \frac{1}{Z}\sum_{a=1}^{N}\sum_{b=1}^{N}\delta(h_{ak}, h_{bk})\sum_{q=1}^{M}\mathbb{1}_k(q)\frac{\mathbb{1}_j(a,b)\cap\delta(y_{aj}, y_{bj})}{\mathbb{1}_j(a,b)}, \quad (28)$$

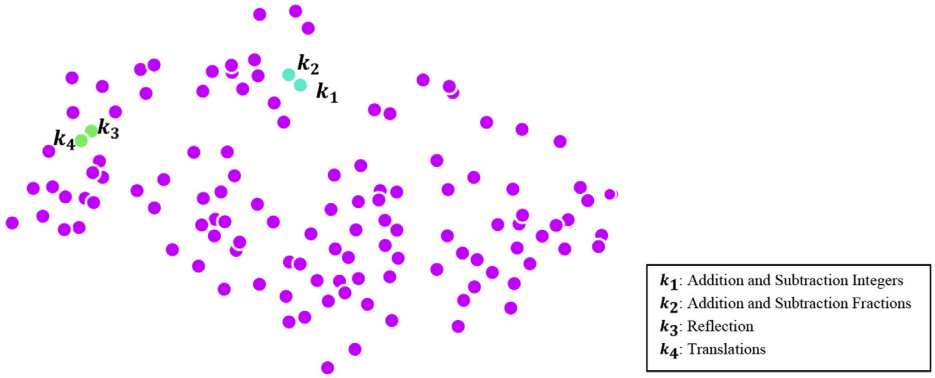


Fig. 9. The knowledge nodes embeddings of GCDM reduced dimension by t-SNE in ASSIST0910. k_1: Addition and Subtraction Integers; k_2: Addition and Subtraction Fractions; k_3: Reflection; k_4: Translations.

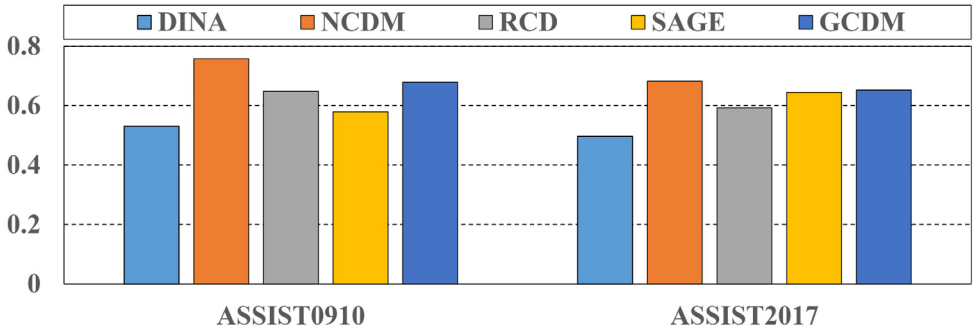


Fig. 10. DOA results of models on both datasets.

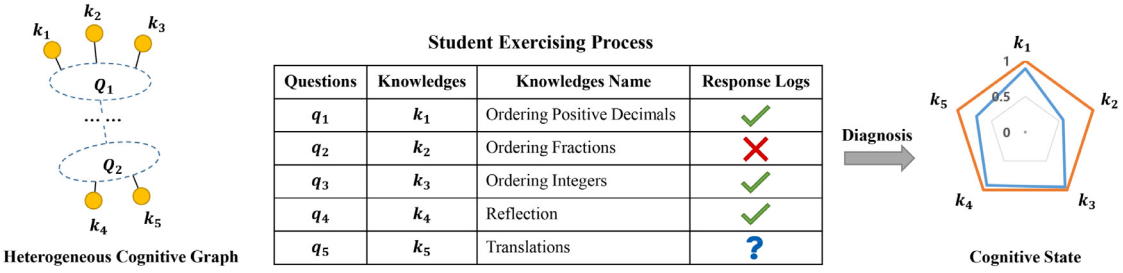


Fig. 11. An example of a Cognitive Diagnosis process in ASSIST-0910.

where $Z = \sum_{a=1}^{N}\sum_{b=1}^{N}\delta(h_{ak}, h_{bk})$. h_{ak} is the cognitive states of student a on the knowledge concepts k as Eq. (16). $\delta(x, y)$ is an indicator function, where $\delta(x, y) = 1$ if $x > y$. $\mathbb{1}_k(q)$ is another indicator function, where $\mathbb{1}_k(q) = 1$ if the questions q contains knowledge concept k. Furthermore, we average DOA(k) of all concepts as DOA to measure the overall performance on the all knowledge concepts, which is denoted as $DOA = \sum_{k=1}^{K} DOA(K) \in [0.0, 1.0]$, the larger DOA, the better performance on ranking performances.

As [18] suggested, among traditional models, IRT and MIRT, there is no clear correspondence between their latent features and knowledge concepts, we, therefore, compare with DINA, NCDM, and RCD which adopt the Q-matrix and the most solid graph-based method, SAGE. Fig. 10 shows the performances on DOA. From the table, we can observe the NCDM has the best performances on DOA. It shows the NCDM where Q-matrix is explicitly introduced and relatively simple neural networks are used to maintain fairly high interpretability. While among the methods with the graph structure, we find the GCDM gets the second level DOA on both datasets. It shows among the methods, even without Q-matrix, GCDM can better mine the high order relation between student nodes and knowledge nodes. By more reasonable mining of the topology structure in the cognitive graph, the representation of the student nodes is more concrete instead of a kind of inexplicable abstract representation, which fully reflects the cognitive degree of students on different knowledge concepts. Besides, compared with the RCD, we can find the larger the cognitive graph, that is the graph from ASSIST2017, our GCDM may relatively better learn the information on the graph. This is because in more complex cognitive graph, the higher-order relationships of nodes based on cognitive and topological relationships in the graph structure are more prominent, and the advantages of our method are more obvious.

Case Study. In Fig. 11, we show an example of applying our model to obtain a diagnostic result of a student from the ASSIST0910 dataset. In this case, the student has finished the first four questions out of $q_1 \sim q_5$, and we need to estimate whether the student can answer the last question correctly. The knowledge concepts of the first three questions k_1, k_2, k_3 (Ordering Positive Decimals, Ordering Fractions, Ordering Integers) and those of the last two questions k_4, k_5 (Reflection, Translations) are two sets of similar knowledge groups. These skills (within each group) are

conceptually similar and connected to a set of related questions. We then reveal the latent capabilities (cognitive states) of the student on different knowledge concepts, where the orange curve indicates the maximum capability (i.e., giving a correct answer with a predicted possibility of 1.0) and the blue curve shows the predicted capabilities of the chosen student. In the figure, we can find that among different knowledge concepts, the concepts with high cognitive states tend to be the concepts that students score high. While the GCDM considers that the student may perform similarly in similar knowledge concepts. So that since the student performs well on similar knowledge concepts, an accidental wrong answer on k_2 will not excessively reduce the model's estimate of his cognitive state on this knowledge concept. Besides, even if the student has not practiced the questions corresponding to k_5, the GCDM gives a positive estimate of a student's cognitive states concerning the performances of the relevant knowledge points k_4. The posterior results show that the student later obtains a high score on this knowledge concept. This shows that the cognitive states h of the students diagnosed are consistent with the performances of the student, and are related to the internal correlation between knowledge concepts, which are explainable.

6. Conclusion

In this paper, we focused on the cognitive diagnosis on the heterogeneous cognitive graph and proposed a novel Graph-based Cognitive Diagnosis Model (GCDM). Specifically, we first built the heterogeneous cognitive graph, where the students, questions, and knowledge concepts are individual nodes and the interactions between students and questions and correlations between questions and knowledge concepts link these nodes. Then we designed the graph-based learning modules, including performance-relative propagator and attentive Knowledge aggregator to infer and update the cognitive states on the graph. Finally, extensive experiments on real-world datasets clearly showed the effectiveness and extendibility of our GCDM. We hope this work could lead to further studies.

CRediT authorship contribution statement

Yu Su: Conceptualization, Methodology, Project administration. **Zeyu Cheng:** Methodology, Investigation, Writing – review & editing. **Jinze Wu:** Methodology, Software, Writing – original draft. **Yanmin Dong:** Software, Writing – original draft. **Zhenya Huang:** Supervision, Resources. **Le Wu:** Supervision. **Enhong Chen:** Supervision. **Shijin Wang:** Funding acquisition, Supervision. **Fei Xie:** Supervision.

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 research was supported/partially supported by Grants No. 2020B444 and the National Natural Science Foundation of China (Grants No. 61976001, 61922073, 61672483).

References

[1] O. Luaces, J. Díez, A. Alonso-Betanzos, A. Troncoso, A. Bahamonde, Content-based methods in peer assessment of open-response questions to grade students as authors and as graders, Knowl.-Based Syst. 117 (2017) 79–87.
[2] A. Anderson, D. Huttenlocher, J. Kleinberg, J. Leskovec, Engaging with massive online courses, in: Proceedings of the 23rd International Conference on World Wide Web, 2014, pp. 687–698.
[3] L. Pappano, The year of the MOOC, N.Y. Times 2 (12) (2012) 2012.
[4] J.R. Anderson, C.F. Boyle, B.J. Reiser, Intelligent tutoring systems, Science 228 (4698) (1985) 456–462.
[5] H. Burns, C.A. Luckhardt, J.W. Parlett, C.L. Redfield, Intelligent Tutoring Systems: Evolutions in Design, Psychology Press, 2014.
[6] J.L. Templin, R.A. Henson, Measurement of psychological disorders using cognitive diagnosis models, Psychol. Methods 11 (3) (2006) 287.
[7] Q. Liu, R. Wu, E. Chen, G. Xu, Y. Su, Z. Chen, G. Hu, Fuzzy cognitive diagnosis for modelling examinee performance, ACM Trans. Intell. Syst. Technol. (TIST) 9 (4) (2018) 1–26.
[8] D. Benyon, D. Murray, Adaptive systems: from intelligent tutoring to autonomous agents, Knowl.-Based Syst. 6 (4) (1993) 197–219.
[9] M.R. Novick, The axioms and principal results of classical test theory, J. Math. Psych. 3 (1) (1966) 1–18.
[10] R.F. DeVellis, Classical test theory, Med. Care (2006) S50–S59.
[11] H. Gulliksen, Theory of Mental Tests, Routledge, 2013.
[12] B.B. Ellis, A.D. Mead, Item Analysis: Theory and Practice using Classical and Modern Test Theory, Blackwell Publishing, 2002.
[13] F. Drasgow, C.L. Hulin, Item Response Theory, Consulting Psychologists Press, 1990.
[14] F.M. Lord, M.R. Novick, Statistical Theories of Mental Test Scores, IAP, 2008.
[15] J. De La Torre, DINA model and parameter estimation: A didactic, J. Educ. Behav. Stat. 34 (1) (2009) 115–130.
[16] J. De La Torre, J.A. Douglas, Model evaluation and multiple strategies in cognitive diagnosis: An analysis of fraction subtraction data, Psychometrika 73 (4) (2008) 595.
[17] J. De La Torre, J.A. Douglas, Higher-order latent trait models for cognitive diagnosis, Psychometrika 69 (3) (2004) 333–353.
[18] F. Wang, Q. Liu, E. Chen, Z. Huang, Y. Chen, Y. Yin, Z. Huang, S. Wang, Neural cognitive diagnosis for intelligent education systems, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 6153–6161.
[19] W. Gao, Q. Liu, Z. Huang, Y. Yin, H. Bi, M.-C. Wang, J. Ma, S. Wang, Y. Su, RCD: Relation map driven cognitive diagnosis for intelligent education systems, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 501–510.
[20] M.M. Keikha, M. Rahgozar, M. Asadpour, Community aware random walk for network embedding, Knowl.-Based Syst. 148 (2018) 47–54.
[21] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowl.-Based Syst. 151 (2018) 78–94.
[22] W.L. Hamilton, R. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.
[23] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.
[24] C. Zhang, D. Song, C. Huang, A. Swami, N.V. Chawla, Heterogeneous graph neural network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 793–803.
[25] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P.S. Yu, Heterogeneous graph attention network, in: The World Wide Web Conference, 2019, pp. 2022–2032.
[26] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, Y. Li, Metapath-guided heterogeneous graph neural network for intent recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2478–2486.
[27] C. Wu, F. Wu, Y. Huang, X. Xie, User-as-graph: User modeling with heterogeneous graph pooling for news recommendation, in: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, 2021, pp. 1624–1630.
[28] H. Ji, J. Zhu, X. Wang, C. Shi, B. Wang, X. Tan, Y. Li, S. He, Who you would like to share with? A study of share recommendation in social e-commerce, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 232–239.
[29] H. Nakagawa, Y. Iwasawa, Y. Matsuo, Graph-based knowledge tracing: modeling student proficiency using graph neural network, in: 2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI), IEEE, 2019, pp. 156–163.
[30] T.A. Ackerman, M.J. Gierl, C.M. Walker, Using multidimensional item response theory to evaluate educational and psychological tests, Educ. Meas.: Issues Pract. 22 (3) (2003) 37–51.
[31] M.D. Reckase, Multidimensional item response theory models, in: Multidimensional Item Response Theory, Springer, 2009, pp. 79–112.

[32] J. Zhang, Y. Mo, C. Chen, X. He, GKT-CD: Make cognitive diagnosis model enhanced by graph-based knowledge tracing, in: 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–8.
[33] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710.
[34] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.
[35] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, S. Jaiswal, Graph2vec: Learning distributed representations of graphs, 2017, arXiv preprint arXiv:1707.05005.
[36] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. 20 (1) (2008) 61–80.
[37] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? 2018, arXiv preprint arXiv:1810.00826.
[38] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European Semantic Web Conference, Springer, 2018, pp. 593–607.
[39] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, D. Yin, Graph neural networks for social recommendation, in: The World Wide Web Conference, 2019, pp. 417–426.
[40] Z. Guo, H. Wang, A deep graph neural network-based mechanism for social recommendations, IEEE Trans. Ind. Inf. 17 (4) (2020) 2776–2783.
[41] A. Chaudhary, H. Mittal, A. Arora, Anomaly detection using graph neural networks, in: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), IEEE, 2019, pp. 346–350.
[42] A. Protogerou, S. Papadopoulos, A. Drosou, D. Tzovaras, I. Refanidis, A graph neural network method for distributed anomaly detection in IoT, Evol. Syst. 12 (1) (2021) 19–36.
[43] X. Song, J. Li, Y. Tang, T. Zhao, Y. Chen, Z. Guan, Jkt: A joint graph convolutional network based deep knowledge tracing, Inform. Sci. 580 (2021) 510–523.
[44] X. Song, J. Li, Q. Lei, W. Zhao, Y. Chen, A. Mian, Bi-CLKT: Bi-graph contrastive learning based knowledge tracing, 2022, arXiv preprint arXiv:2201.09020.
[45] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, 2013, arXiv preprint arXiv:1301.3781.
[46] A. Ghosh, N. Heffernan, A.S. Lan, Context-aware attentive knowledge tracing, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2330–2339.
[47] S. Pandey, J. Srivastava, Rkt: Relation-aware self-attention for knowledge tracing, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1205–1214.
[48] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.
[49] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[50] S.E. Embretson, S.P. Reise, Item Response Theory, Psychology Press, 2013.
[51] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[52] J. Fogarty, R.S. Baker, S.E. Hudson, Case studies in the use of ROC curve analysis for sensor-based estimates in human computer interaction, in: Proceedings of Graphics Interface 2005, 2005, pp. 129–136.
[53] Q. Liu, Z. Huang, Y. Yin, E. Chen, H. Xiong, Y. Su, G. Hu, Ekt: Exercise-aware knowledge tracing for student performance prediction, IEEE Trans. Knowl. Data Eng. 33 (1) (2019) 100–115.
[54] K.H. Wilson, X. Xiong, M. Khajah, R.V. Lindsey, S. Zhao, Y. Karklin, E.G. Van Inwegen, B. Han, C. Ekanadham, J.E. Beck, et al., Estimating student proficiency: Deep learning is not the panacea, in: Neural Information Processing Systems, Workshop on Machine Learning for Education, 2016, p. 3.
[55] Y. Su, Q. Liu, Q. Liu, Z. Huang, Y. Yin, E. Chen, C. Ding, S. Wei, G. Hu, Exercise-enhanced sequential modeling for student performance prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.
[56] L. Van der Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (11) (2008).
[57] F. Fouss, A. Pirotte, J.-M. Renders, M. Saerens, Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation, IEEE Trans. Knowl. Data Eng. 19 (3) (2007) 355–369.
[58] Z. Huang, Q. Liu, Y. Chen, L. Wu, K. Xiao, E. Chen, H. Ma, G. Hu, Learning or forgetting? A dynamic approach for tracking the knowledge proficiency of students, ACM Trans. Inf. Syst. (TOIS) 38 (2) (2020) 1–33.