PreBiGE: Course Recommendation Using Course Prerequisite Relation Embedding and Bipartite Graph Embedding

Hafsa Kabir Ahmad[1,2], Bo Liu[2], Bello Ahmad Muhammad[1,2], Mubarak Umar [1,2]
[1] Department of Computer Science, Bayero University, Kano 700241, Nigeria
[2]School of Computer Science, Shaanxi Normal University, Xi'an 710062, China

A growing number of students enrol in online education to improve their skills. However, students are faced with the challenge of finding courses that meet their individual needs. Recommender systems were introduced to help students choose the courses that best meet their needs. To learn better representations of students and courses for improved recommendation results, existing graph-based recommender systems utilize the high-order collaborative signals between set of students or set of courses from a bipartite graph. However, courses also have prerequisite dependency between them, which when utilized together with collaborative relations can improve recommendation results. On this basis, we propose a model that utilizes the high-order relation between set of courses, the prerequisite dependency between courses, as well as the direct relation between students and courses. Using meta-paths generated from the knowledge graph, our model extracts the prerequisite dependency between courses, which is then used to generate a course prerequisite graph. The course prerequisite graph and the student-course bipartite graph are used to learn the representation of the students and courses, jointly capturing the prerequisite dependency, high-order collaborative relations as well as direct relations. The learned representations are used for recommendation. The experiments on real-world dataset show the superiority of our proposed method, achieving 3.61% on F1@10 and 1.38% on Mrr@10.

***Index Terms*—course recommendation, prerequisite dependency, MOOCs, knowledge graph**

I. Introduction

The Covid-19 pandemic has affected modes of teaching leading to an increase in more independent online learning platforms [1]. Although students enrolling on these learning platforms have access to abundant learning resources, they can easily become overwhelmed by the volume of information. Therefore, recommender systems have become an effective tool to help students obtain the appropriate resources to meet their personal learning goals.

The most popular type of recommender systems is the collaborative filtering (CF) techniques. In numerous domains, many CF-based recommender systems have been implemented (e-commerce, poi, articles etc). CF-based RS implies that users who do similar things are likely to like the same things (User-based CF) or that users will like items similar to what they previously liked (Item-based CF)[2]. Model-based CF learns the users' and items' representation from the user-item interaction matrix. Existing model-based course RS [3, 4, 5] utilized information from external sources to learn better representations of students and courses. Graph-Based RS [6, 7, 8] have proven that utilizing the high-order student-student and course-course can generate better students' and courses' representations thereby improving the recommender system. Although the high-order collaborative relations between courses are explored in existing works, they fail to utilize the prerequisite dependencies that are between courses. These prerequisite relations between courses represent the dependency among different courses. For example, two courses with prerequisite dependency $< c_1, c_2 >$ indicate that for a student to learn c_2 he/she has to learn c_1 first. Therefore, introducing the prerequisite dependency between courses is needed to improve the effectiveness of a recommender system.

To address the above-mentioned challenge, we present a novel model, PreBiGE short for prerequisite and bipartite graph embedding for course recommendation. Our model captures the prerequisite dependency between courses, the high-order collaborative relations between set of course and set of students, and the direct relations between students and courses to learn their high-quality representations for better recommendation results. The course prerequisite dependency is generated from a knowledge graph. Specifically, we create a knowledge graph containing entities (student, course, concept) and relations (student-course, course-concept,concept-concept prerequisite relation) to infer the course-course prerequisite relation. Then, we utilize the direct relation between the students and courses, high-order collaborative relationships between the students set and courses set, and the prerequisite dependency between courses. The representations of courses learned from high-order collaborative relations and the prerequisite relations between them are fused to learn a better representation of courses.

We summarize the contributions as follows:

1) We propose a method for inferring the course level prerequisite relations from a knowledge graph.
2) We present a method for recommending courses to students based on the fusion of high-order collaboration and prerequisite relations of courses, the relation between students and the direct relations between students and courses.
3) We conduct extensive experiments to evaluate the performance of PreBiGE with existing recommendation meth-

Manuscript received November 4, 2022; revised December 26, 2022.
Corresponding author (s): Bo Liu (liubojsj@snnu.edu.cn), Hafsa Kabir Ahmad (hkahmad.cs@buk.edu.ng)
Email Addresses: Bello Ahmad Muhammad(bamuhammad@snnu.edu.cn), Mubarak Umar (mubarakumar@snnu.edu.cn)
This work is supported by the Major Achievements Cultivation Project of the Collaborative Innovation Center of Assessment toward Basic Education Quality at Beijing Normal University (2020-05-0034-BZPK01).

ods.

The remainder of this work is divided into the following sections. Section II briefly discusses the related works. We present a preliminary of the proposed work in section III. We define the methodology of the proposed work in Section IV. The experiment and its results are presented in Section V, along with discussions. Section VI gives the conclusion of our work.

II. Related Works

In this section, we present existing works on course recommender systems and prerequisite relation extraction in online education

A. Course Recommender Systems in Online Education

Collaborative filtering (CF) recommender systems are the most widely used recommender systems in MOOCs. It predicts which course a student may interact with based on his/her historical interactions.

User-Based CF assumed that a student would have the same course enrollment as the students that have similar past enrollments while Item-Based CF assumed that students would prefer courses that are similar to those they have previously enrolled in. Existing User-based CF utilized information from external sources. S. Dias *et al.* [9] utilized the social information of students to identify those with the highest degree of similarity. Jing and Tang [10], Yang *et al.* [11] utilized content information in addition to the User-Based CF for improved recommendation results. Item-based recommender system [2] recommended courses based on the collaborative relations between courses and their prerequisite dependencies.

Model-Based CF utilized machine learning techniques to learn users' and items' representations from their interaction matrix. However model-based CF learned poorer representations if the user-item interaction is sparse. To address this challenge, Symeonidis and Malakoudis [3] utilized side information such as external matrices and Li *et al.* [4] utilized social influence. Zhang *et al.* [5] applied deep learning techniques to mine students and course features. With the advent of graph learning techniques, [8] and Ahmad *et al.*, [6] utilized the high-order collaborative signal to address the sparsity challenge. Although the high-order collaborative relation is used to improve the quality of course recommendations, the systems failed to consider the prerequisite dependencies between courses.

B. Prerequisite Relation Extraction

Prerequisite relations in online education have been inferred using different techniques in various studies. Roy *et al.* [12] proposed a supervised learning approach to infer concept prerequisite relation. The prerequisite relation between concepts is inferred from the course prerequisite relations and the labelled concept prerequisite data generated from the concept's representation, which is derived from a pairwise latent dirichlet allocation model. Liu *et al.* [1] proposed a deep learning based concept prerequisite dependency prediction model. The distance between the courses vector representations learned in the hyperbolic space using word embedding was calculated, which was further used to determine their prerequisite relationship through hyperbolic neural networks. Pan *et al.* [13] generated the latent representation between concepts using representation learning-based method. Then utilized the semantic relatedness between concepts, the contextual feature as well as the structural features to help infer prerequisite relations. Inspired by the work of Pan *et al.* [13], Zhao *et al.* [2] predicted the prerequisite relation between concepts from features such as the position of concept in a course, the distribution of the feature, the appearance of concepts in more videos and its duration as well as the co-occurrence of concepts using a random forest classifier. They further extracted the course prerequisite relation from the concept relations generated. Because of the unavailability of labelled data, Liang *et al.* [14] proposed an active learning approach to predicting prerequisite relations. In [15], Yu *et al.*, generated the prerequisite relation by training labelled prerequisite relations generated from concept taxonomy information and video dependency.

III. Preliminary

In this section, we first formulate the problem of course recommendation in MOOCs. To simplify the presentation, we present some key definitions and notations used in the paper. Table I shows the notations that were used.

A. Task Formulation

Given the set of students $U = \{u_1, u_2, ..., u_{|U|}\}$ and the set of courses $C = \{c_1, c_2, ..., c_{|C|}\}$ and the prerequisite relations inferred from KG. Let $E = (u, c) \mid u \in U, c \in C$ indicate the interactions between the students and the course. Our approach aims to learn the embedding of students and courses from the direct relations between them, the high-order collaborative relations and the prerequisite dependency between courses inferred from the KG. Each student is recommended Top-N courses based on the resulting high-quality representations.

B. Definition of Terms

Definition 1 (Bipartite Graph): An student-course interaction graph (bipartite graph) $G = (U, C, E)$ is a graph containing students' set U and courses' set C as nodes. The edge of the graph $E \subseteq U \times C$ denotes the direct relation between them (Student-Course interaction data).

Definition 2 (Bipartite Graph Embedding): The task of bipartite graph embedding is to learn the representations of the nodes in the graph using a mapping function:

$$f : U \cup C \to \mathbb{R}^d$$

. The mapping function ensures that the direct relations between U and C are preserved as well as the high-order collaborative relations between nodes U and nodes C.

Definition 3 (High-Order Collaborative Relation): The high-order collaborative relation refers to the indirect relation between set of courses or set of students. It is determined using the transition probability in a random-walk generated

TABLE I. Notations used and their descriptions.

Notations	Description
G	Bipartite Graph
G_U	Students' Homogeneous Graph
G_C	Courses' Homogeneous Graph
G_{CP}	Courses' Prerequisite Relation Graph
U, C, K	Set of students, courses, and knowledge concepts, respectively
$E \subseteq U \times C$	Edge set in G
A_u,A_c, A_{cp}	Adjacency matrix of G_U, G_C and G_{CP}
$< k_a, k_b >$	Prerequisite dependency between concept pair
$< c_i, c_j >$	Prerequisite dependency between course pair
$MP(c_i, c_j) = \{mp_1, \ldots, mp_s\}$	Paths between course pair
$mp(c_i, c_j) = n_i \rightarrow n_2 \ldots \rightarrow n_j$	A path between course pair
u_i,c_j	student and course
$u_i{}'$,$c_j{}'$	Embedding vector of student u_i and course v_j

sequence. This is represented as the matrix $H = R^1 + R^2 + R^3 + ... + R^s$, where R^1 denote the normalized direct relation matrix. R^k denotes the normalized s-step probability transition matrix. $h_{ij} \in H$ denote the high-order collaborative relations between node i and node j.

Definition 4 (Concept Prerequisite Relation): This is considered as the dependency between two concepts. Given two concepts k_1 and k_2 from concept set K, students must understand k_1 before learning k_2. Thus, we consider that $< k_1, k_2 >$ represents a prerequisite relationship between k_1 and k_2.

Definition 5 (Course Prerequisite Relation): Similar to the concept prerequisite relation, it describes the prerequisite dependency between two courses. We will infer the course prerequisite relation from the concept prerequisite relation using a knowledge graph. For example, we may infer prerequisite relations such as $c_1, c_2 >$ indicating that for a student to learn c_2 he/she has to learn c_1 first.

Definition 6 (Knowledge Graph): Let $N = \{n_1, n_2, ..., n_k\}$ represent the set of nodes and $R = \{r_1, r_2, ..., r_k\}$ represent set of nodes relations, a knowledge graph is a directed graph $G = (N, L)$ with a node type mapping function $\psi : N \rightarrow A$ and a link type mapping function $\lambda : L \rightarrow R$. Each node $n \in N$ is a node type $\psi(n) \in A$, and each link $l \in L$ is a link type $\lambda(l) \in R$. Because there are multiple types of nodes and node relationships i.e., $|N| > 1$ and/or $|R| > 1$ in the KG used in this study, it can be regarded as a heterogeneous information network. Figure 1 provides a toy example of KG with nodes types (courses and knowledge concept) and links describing the node relations such as (e.g., "contain in", "prerequisite of").

Definition 7 (Meta-Path): Semantic meta-path refers to a sequence of entities connected by relations between two nodes u_i and v_j, which can be represented as $mp(u_i, v_j) = n_0 \xrightarrow{r_1} n_1 \xrightarrow{r_2} ... \xrightarrow{r_k} n_k$. We use $MP(n_i, n_j) = \{mp_1, mp_2, ...mp_{|mp|}\}$ to represent the connected paths between nodes e_i and e_j where mp is a path and $|mp|$ denotes the total number of paths. Consider the KG in Figure **??**, we can infer the prerequisite relation between two courses using meta-paths:

IV. Proposed Methodology

In this section, we present a model that learns the representations of courses from their direct relations, collaborative relations and the prerequisite relations generated from KG. Our proposed model consists of three components: 1) Extracting course prerequisite relations from KG and prerequisite graph construction. 2) Learning the representations of students and courses from their high-Order collaborative relations, direct relations and course prerequisite relations. 3) Course recommendation using the learned representations. The subsequent subsections provide a detailed discussion of our proposed methodology. The algorithm of our proposed method is given in Algorithm 1.

A. Course Prerequisite Relations Extraction and Prerequisite Graph Construction

In this section, we start by extracting the concept-concept prerequisite pair which is further used together with other relations to infer course prerequisite relations from knowledge graph. The course prerequisite relation is then used to construct the course prerequisite graph.

1) Concept Prerequisite Relation Extraction

Given a set of courses $C = \{c_1, c_2, ..., c_{|C|}\}$, set of videos contained in courses $V = \{v_1, v_2, ..., v_{|V|}\}$ and a set of knowledge concepts taught in each video $K = \{k_1, k_2, ..., k_{|K|}\}$, we adapt the process used by [15] to extract the prerequisite relations between concepts. In each video, the 10 most representative course concepts are extracted from subtitles. Each concept's description is extracted from Wikidata and top 10 related papers through Aminer[1]. Word embedding is used to calculate the most likely category of a course concept. Some annotators are asked to label if the concept belongs to a category. For concept pairs that are labelled "not belong to" the brother category of the prior one is chosen as a new candidate and put the refreshed pair into the annotation pool. This results in a reduced annotation pool with valid concept pair. Given the concept annotation pool, labelling all possible concept pairs is infeasible. Thus, a sample of candidate concept pair is created only for concepts which occur in the same course. Annotators label the candidate course pairs. Then a model is trained to label the unlabelled course pairs. Another classifier is trained to give labelled pairs with low confidence score a new label. This result in a labelled concept pairs indicating if concept k_a is helpful to understand concept k_b

[1]https://aminer.org

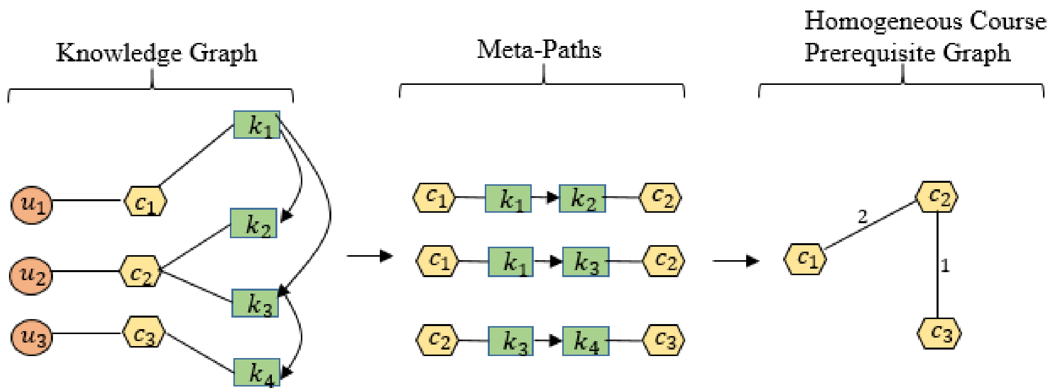


Fig. 1. Course Prerequisite Relations Extraction

These descriptions are used to create a concept taxonomy. Candidate concept pairs are generated by selecting concepts which belong to the same course. The concept taxonomy information is utilized to reduce the candidate concept pair. Some candidates are then manually labelled by annotators to identify if concept X is important in understanding concept B. This result in some labelled concept pair. Then a model is trained to classify the other unlabeled pairs. This generates concept prerequisite pairs $< k_a, k_b >$.

2) Extracting Course Prerequisite Relation from KG

Given the prerequisite relations between concepts $< k_a, k_b >$ generated from the previous section IV-A1, indicating that k_2 is a follow-up concept of k_1, the course-concept relation, indicating that a concept is contained in a course, and the student-course relations, we construct a directed knowledge graph containing students nodes U, courses nodes C, concepts nodes K and the relations between them (student-course R, course-concept C_K, concept-concept K_K).

Based on the constructed knowledge graph, we select different meta-paths between a course pair. The meta-path between course pair is given as $mp(c_i, c_j) = n_i \rightarrow n_2 \ldots \rightarrow n_j$. After generating all meta-paths between course pairs given as $MP(c_i, c_j) = \{mp_1, \ldots, mp_s\}$. If $lenght(MP(c_i, c_j)) \neq 0$, then we infer that there is a prerequisite relation $< c_i, c_j >$ between them. For example, in Figure 1, we have meta-path between c_1 and c_2, $c_1 \rightarrow k_1 \rightarrow k_2 \rightarrow c_2$. This indicates that there exists a prerequisite relation between k_1 and k_2. With k_1 being a concept in c_1 and k_2 a concept in c_2, we can infer that c_1 is a prerequisite of c_2. In addition, we count the number of paths that are between each course pair to differentiate the strength of their relation i.e., $lenght(MP(c_i, c_j)$. This number will serve as the weight of the edge in the course prerequisite graph to be constructed in the next section.

3) Course Prerequisite Graph Construction

Given the prerequisite relations between courses pairs generated from section IV-A2, we construct a weighted homogeneous graph. The weights of the edges between the nodes of the course prerequisite graph indicate the number of paths that are between the connected course nodes. The matrix $\mid CP \mid \times \mid CP \mid$ matrix of $A_{cp} = [a_{ij}^{CP}]$ are used to represent the adjacency matrix of the prerequisite graph.

B. Direct Relation, High-Order Collaborative Relations and Prerequisite Graph Embedding

Given the student-course interaction data, we first construct a bipartite graph $G = (U, C, E)$. The weight matrix of the graph is denoted as A. A good graph embedding should be able to preserve all the relations that are in the graph. We utilize the direct student-course relation as well as the high-order relations between students' set and the high-order relations between courses' set. We consider jointly learning the representations of the students and courses by preserving these relations in the bipartite graph and the course prerequisite graph.This section is divided into: 1) Direct relation embedding and 2) High-order collaborative relations embedding and prerequisite relation embedding.

1) Direct Relation Embedding

The goal is to learn the representations (embedding) of the students and courses that preserves the direct relation between them. That is, if there is an edge between them, then the representation should encode the edge. To preserve the direct relation, we utilize first-order proximity in LINE [16]. We model the direct relations between students and courses by considering the direct proximity between them. The joint probability between student u_i and course j is defined as:

$$P(i, j) = \frac{a_{ij}}{\sum_{e_{ij} \in E} a_{ij}} \tag{1}$$

where a_{ij} denote the weight of edge e_{ij}. Whenever the weight is large, then the two nodes have a higher probability to co-occur. To estimate the direct proximity in the embedding space, we use the inner product to model the interaction between two nodes inspired by Word2Vec. We use a sigmoid function to transform the interaction into probability space:

$$\hat{P}(i, j) = \frac{1}{1 + exp(-u_i'^{T} c_j')} \tag{2}$$

where u_i' and c_j' represent the embedding matrices of nodes u and c respectively. Our goal is now to minimize the difference between the probability of the vertices and their reconstructed embedding. We choose KL-divergence

$$min\ L_1 = KL(P||\hat{P}) = \sum_{e_{ij}\in E} P(i,j) log(\frac{P(i,j}{\hat{P}(i,j)}$$
$$\infty - \sum_{e_{ij}\in E} a_{ij} log \hat{P}(i,j) \quad (3)$$

$\sigma(\cdot)$ is the sigmoid function.

2) High-Order Collaborative Relations Embedding and Prerequisite Relation Embedding

Embedding high-order collaborative relations in recommendation has proven to be effective [17, 18]. This motivates us to explore the high-order collaborative relations in bipartite graph. In addition, exploring the prerequisite relation has also been effective in improving recommendation results [2]. Although exploring the direct relations can recover the high-order relations, it is impractical to rely on them as most direct relations are sparse in real life. As such, we speculate that embedding the high-order relations as well as the prerequisite relations could bring extra benefits to the direct relation embedding.

To explore the high-order collaborative relations, we resort to the solution of DeepWalk [19]. Specifically, the bipartite graph is split into two corpora of node sequences by performing random walks; then the embeddings are learned from the corpora to represent the high-order relations between nodes. To explore the prerequisite relations, we also generate the corpus of node sequences by performing random walks on the prerequisite graph; then learn the embedding from the corpus.

- Constructing Corpus of Node Sequences: Homogenous graphs such as the prerequisite relations graphs can be converted into a corpus of node sequences by performing random walks on the graphs. However, since the distribution of random walks on bipartite graphs is not stationary, directly performing the walks on student-course bipartite graph could fail. To address this issue, we generate two collaborative relations graphs (homogeneous) between students' nodes and courses' nodes from the student-course bipartite graph. To find the collaborative relations, we employ the concept of Co-HITS [20]. This captures the second-order proximity between the nodes thereby projecting the bipartite graph into two homogeneous graphs.

$$a^U_{i_j} = \sum_{k\in C} a_{ik} a_{jk}\ ;\ a^C_{ij} = \sum_{k\in U} a_{ki} u_{kj} \quad (4)$$

  where a_{ij} is the weight of the edge. Therefore, the $\mid U \mid \times \mid U \mid$ matrix of $A_u = [a^U_{ij}]$ and the $\mid C \mid \times \mid C \mid$ matrix of $A_c = [a^C_{ij}]$ represent, respectively, the adjacency matrices of the two projected homogeneous networks.

  Now we can generate two corpora for learning the high-order collaborative relations by doing random walks on the two homogeneous graphs. Additionally, we perform random walks on the prerequisite relation graph. We employ weighted random walks and negative sampling a corpus of node sequences for each of the reconstructed graphs and the prerequisite homogeneous graph [21]. The node sequences, $walks_u$, $walks_c$ and $walks_{cp}$, are generated in Algorithm 1, lines 10-22.
- High-order Collaborative Relations and Prerequisite Relations Embedding: After performing random walks on the three homogeneous graphs, we obtain three corpora of node sequences each for the student collaborative relations, courses collaborative relations and courses prerequisite relations, respectively. We then employ the Skip-gram model [22] on the three corpora to learn node embeddings. The goal is to capture the high-order relations in the graph, implying that similar representations should be given to nodes sharing context nodes in the sequences. The context is the ws nodes before node u_i and after node u_i in node sequence S. The same applies to node c_j. Another node sequence from the prerequisite graph, SC, is connected to the node c_j. The node c_j is also associated with another node sequence SC from the prerequisite graph. Therefore, it is also associated with ws context nodes before c_j and after c_j in SC. Each node is associated with a context vector θ_i (or Θ_j or ϑ_j) to denote its role as a context. For each node c_j in C, we preserve both its high-order collaborative relations and the prerequisite relation. For each node u_i in U, we preserve only its high-order collaborative relations. For the corpus $walks_u$,$walks_c$, and $walks_{cp}$ we maximize the conditional probability:

$$\max\ M_2 = \prod_{u_i\in S\wedge S\in walks_u} \prod_{u_d\in D_S(u_i)} P(u_d \mid u_i) \quad (5)$$

where $D_S(u_i)$ indicates the node u_i's context nodes in sequence S. We also get the objective vector of each node type in $walks_c$

$$\max\ M_3 = \prod_{c_j\in S\wedge S\in walks_c} \prod_{c_d\in D_S(c_j)} P(c_d \mid c_j) \quad (6)$$

We then get another objective vector of node c_j using $walks_{cp}$. $D_{SC}(c_j)$ denotes the context nodes of node c_j in sequence SC.

$$max\ M_4 = \prod_{c_j\in SC\wedge SC\in walks_{cp}} \prod_{c_{dc}\in D_{SC}(c_j)} P(c_{dc} \mid c_j) \quad (7)$$

The conditional probability is parameterize $P(u_d \mid u_i)$, $P(c_j \mid c_j)$ and $P(c_{dc} \mid c_j)$ using softmax for output [21, 19]:

$$P(u_d \mid u_i) = \frac{exp({u'_i}^T \theta'_d)}{\sum_{l=1}^{|U|} exp({u'_i}^T \theta'_l)}\ ,$$
$$P(c_d \mid c_j) = \frac{exp({c'_j}^T \Theta'_d)}{\sum_{l=1}^{|C|} exp({c'_j}^T \Theta'_l)}\ ,$$
$$P(c_{dc} \mid c_j)) = \frac{exp({c'_j}^T \vartheta'_{dc}))}{\sum_{l=1}^{|C|} exp({c'_j}^T \vartheta'_l)} \quad (8)$$

where $P(u_d \mid u_i)$ represents the likelihood that u_d would be observed in the contexts of u_i. Same applies to $P(c_d \mid c_j)$ and $P(c_{dc} \mid c_j))$.

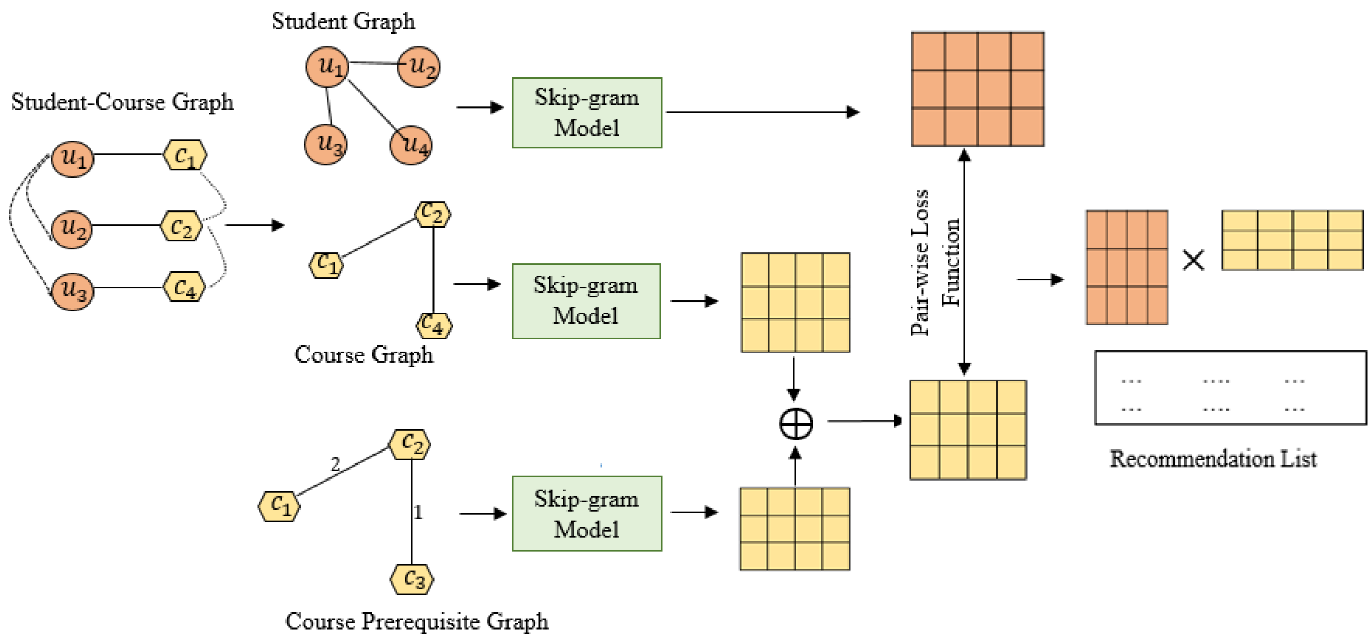


Fig. 2. The Overall Framework of PreBiGE. It consists of the process of learning the representations of students and courses from a student-course bipartite graph and course prerequisite graph using random walk and skip-gram Model. The representations are further used to estimate the missing values in the student-course rating matrix, which is further used to provide a ranked list of recommended courses.

3) Joint Optimization

To preserve the prerequisite relations, high-order collaborative relations and direct relations, we need to optimize an objective function. We adopt the concept of [17] to jointly optimize their objective functions.

$$\max\ L = \alpha logM_2 + \beta(logM_3 + LogM_4) - \gamma M_1 \tag{9}$$

where parameters α, β and γ are hyper-parameters in the joint optimization process specified to combine various components. We use Stochastic gradient ascent (SGA) to optimize the model jointly. Each component in Equation 9 have a different training instance. The SGA is modified as follows:

i) Step 1: To learn the representations of the direct relations, we update the vectors u_i' and c_j' using SGA to maximize the last component $L_1 = -\gamma M_1$ as follows:

$$u_i' = u_i' + \lambda\{\gamma a_{ij}[1 - \sigma({u_i'}^T c_j')] \cdot c_j'\} \tag{10}$$

$$c_j' = c_i' + \lambda\{\gamma a_{ij}[1 - \sigma({u_i'}^T c_j')] \cdot u_i'\} \tag{11}$$

where λ represents the learning rate.

ii) Step 2: To learn the representations of the high-order collaborative relations and course prerequisite relations, we treat nodes u_i, c_j as the centre nodes. Then use SGA to maximize the objective functions $L_2 = \alpha logM_2$, $L_3 = \beta logM_3$ and $L_4 = \beta logM_4$. Given the nodes u_i, c_j and their context vertex u_d, c_d, c_{dc}, we update their corresponding embedding vectors u_i' ,c_j', as follows:

$$u_i' = u_i' + \lambda\{\sum_{z\in\{u_d\}\cup N_S^{ns}(u_i)} \alpha[I(z, u_i) - \sigma({u_i'}^T \theta_z')] \cdot \theta_z')\} \tag{12}$$

$$c_j' = c_j' + \lambda\{\sum_{z\in\{c_d\}\cup N_S^{ns}(c_j)} \beta[I(z, v_j) - \sigma({c_j'}^T \Theta_z')] \cdot \Theta_z')\} \tag{13}$$

Update c_j again as follows:

$$c_j' = c_j' + \lambda\{\sum_{zc\in\{c_{dc}\}\cup N_{SC}^{ns}(c_j)} \beta[I(zc, v_j) - \sigma({c_j'}^T \vartheta_{zc}')] \cdot \vartheta_{zc}')\} \tag{14}$$

where $N_S^{ns}(u_i)$ represents the negative samples ns for a node at the center u_i. The same holds true for $N_S^{ns}(c_j)$ and $N_{SC}^{ns}(c_j)$. $I(z, u_i)$ is a function that assesses whether or not node z is in the context of u_i; $I(z, c_j)$ and $I(zc, c_j)$ have the same meaning. In addition, the positive context vectors and negative context vectors are updated as follows:

$$\theta_z' = \theta_z' + \lambda\{\alpha[I(z, u_i) - \sigma((u_i')^T \theta_z')] \cdot u_i')\} \tag{15}$$

$$\Theta_z' = \Theta_z' + \lambda\{\beta[I(z, c_j) - \sigma((c_j')^T \Theta_z')] \cdot c_j')\} \tag{16}$$

$$\vartheta_{zc}' = \vartheta_{zc}' + \lambda\{\beta[I(zc, c_j) - \sigma((c_j')^T \vartheta_{zc}')] \cdot c_j')\} \tag{17}$$

4) Computational Complexity Analysis

We have introduced PreBiGE Learning Algorithm 1, which includes the following three steps. Firstly, we design a strategyfor extracting prerequisite relations between two courses by generating meta-paths from the knowledge graph (lines 2-8). We then generate corpora node sequences from the three graphs, one from the prerequisite relation and the other two from the students and courses collaborative relations graphs ((lines 13-25). Finally, we jointly learn the representations of students and courses encoding all the relations between them. (26-40).

Algorithm 1 PreBiGE Learning Algorithm

Require: U, C, K, Rating Matrix R, course-concept relations C_k, concept-concept relations K_k, window size w, walks per node r, embedding dimension dim, walk length l
Ensure: Embedding of nodes U and C
Create a knowledge graph KG using U, C, K, R C_k, K_k.
for each (c_i, c_j) pair **do**
    mine connected paths $MP(c_i, c_j)$ from KG
    if $MP(c_i, c_j) \neq 0$ **then**
        Create a prerequisite relation $< c_i, c_j >$
        Count the number p of paths between them
    end if
end for
Create a weighted homogeneous course prerequisite graph G_{cp} using $< c_i, c_j >$ with p as edge weight and with adjacency matrix A_{cp}
Create a Bipartite Graph $G = U, C, E$ with weight matrix A using U, C, R
Generate two homogeneous graph G_U and G_C for U and C, respectively and obtain two adjacency matrix A_u and A_c for each graph using Equation 4
Initialize embedding vectors and context vectors u'_i, c'_j, θ'_i,Θ'_j,ϑ'_j
Initialize a list $walks_u, walks_c, walks_{cp}, \Leftarrow [\,], [\,], [\,]$
for all $u_i \in A$ **do**
    $walk_u \Leftarrow RandomWalk(R, u_i, l)$
    Append $walk_u$ to $walks_u$
end for
for all $v_j \in A$ **do**
    $walk_c \Leftarrow RandomWalk(R, c_j, l)$
    Append $walk_v$ to $walks_v$
end for
for all $c_k \in A_{cp}$ **do**
    $walk_{cp} \Leftarrow RandomWalk(A_{cp}, c_k, l)$
    Append $walk_{cp}$ to $walks_{cp}$
end for
for each $edge(u_i, c_j)$ **do**
    update u'_i and c'_j using Equation 10 and 11
    for each (u_i, u_d) in the sequence $S \in walks_u$ **do**
        Generate $N_S^{ns}(u_i)$ using negative sampling
        Update u'_i and θ'_z using Equation 12 and 15, respectively. where $z \in \{u_d\} \cup N_S^{ns}(u_i)$
    end for
    for each (c_j, u_d) in the sequence $S \in walks_c$ **do**
        Generate $N_S^{ns}(c_j)$ using negative sampling
        Update c'_j and Θ'_z using Equation 13 and 16, respectively. where $z \in \{c_d\} \cup N_S^{ns}(c_j)$
    end for
    for each (c_j, c_{dc}) in the sequence $SC \in walks_{cp}$ **do**
        Generate $N_S^{ns}C(c_j)$ using negative sampling
        Update c'_j and ϑ'_z using Equation 14 and 17, respectively. where $z \in \{c_{dc}\} \cup N_S^{ns}C(c_j)$
    end for
end for
return node embedding of U and C

The time complexity includes three key parts: (1) Generating the meta paths: the time complexity of generating the meta-paths is related to the average degree of the entities $\bar{d}$ in the knowledge graph and the number of courses $|C|$. (2) Corpus Generation: Given the walk length of generating the corpus l, the visitation number of nodes in a graph n and the number of walks per node, the time complexity of generating the corpus of each graph is $O(rln)$. (3) Joint Optimization: Given the generated corpus, suppose the visitation count of node v in the generated corpus is vc. The context size is therefore $vc \cdot 3ws$. The time complexity is $(|E| \cdot bs \cdot 3ws \cdot (ns + 1))$.

C. Top-N Course Recommendation

The previous section generated high-quality embedding for all students and courses. The course embedding encodes the course prerequisite dependency between the courses, the high-order collaborative relations as well as the direct relations between the courses and the students. This results in high-quality embedding that can improve the recommendation result. Also, the students' embedding encodes the high-order collaborative relations as well as the direct relations between the courses and the students. To generate the recommendation result, we calculate the similarity between each student's embedding with all other courses embedding using the dot product. To get the Top-N courses, we sort the similarity result in descending order and choose the first N courses.

V. Experiments and Analysis

We aim to answer the following research questions through experiments:

i) RQ I: How does our model, PreBiGE, compare with state-of-the-art graph embedding methods and graph-based course recommender systems?
ii) RQ II: Does including the course prerequisite relations improve the quality of course recommendation results?
iii) RQ III: How do different strengths of the prerequisite relation affect the recommendation result?
iv) RQ IV: How do hyper-parameters affect the performance of our proposed model?

The experimental setup and answers to the aforementioned research questions are described in the sections that follow.

A. Dataset Description

We used the dataset provided by MOOCCube [15]. It is a dataset collected from XuatengX[2] MOOC platform. The statistics of the used dataset are given in Table II. We select the course interaction behaviour occurring from 1st September 2018 to 10th December 2018 from the dataset. To train the model, we randomly sample 60% of the dataset. We then test the model with the remaining 40%. 10 folds of the train-test split are used to avoid overfitting. The hyper-parameters settings are tuned on the first fold only. The optimal hyper-parameter settings are used and then the average performance of all folds is reported. This method of the train-test split is a valid sampling approach [23].

[2] https://next.xuetangx.com/

TABLE II. Description of the dataset.

Nodes/Relations	Name	Statistics
	Students	884
Nodes	Course	307
	Knowledge Concept	17,614
	student-course	5,921
	course-concept	77,091
Relations	concept-concept	905

B. Experimental Settings

1) Comparison Methods

Our model uses high-order collaborative relations between students' set and courses' set, course prerequisite dependencies, and direct student-course relations to improve course recommendations. To choose the comparison method, we identify the recommendation methods that perform similar tasks. In the field of education, there is limited availability of reference datasets and many recommender systems were based on private institutional datasets [9, 24]. In addition, [25] observed that the accuracy of models varies according to the dataset used. Models with good recommendation results on three datasets may not provide good results in other datasets. Based on this, we select domain-independent models:

1) BPRMF [26]: a baseline recommendation approach that optimizes a pairwise loss function suitable for a ranking-based recommendation.
2) NeuMF [27]: a method that substitutes the inner product's MF function with a nonlinear neural network in a neural collaborative filtering method.
3) NGCF [18]: is a graph convolution network-based method that stacked multiple embedding propagation layers to capture the interaction graph's collaborative signal.
4) LightGCN [28]: simplifies the GCN used in NGCF to make it appropriate for recommendation.
5) BiNE [17]: a bipartite graph embedding method that utilizes the direct relation between users and items and the indirect collaborative relations between users and items in the graph.

2) Evaluation Metrics

Given the recommended list of student R_u and the corresponding ground truth enrollment set of student u in the test set, C'_u, we compare our proposed model's performance to that of existing models using the following evaluation metrics:

i) Mean Reciprocal Rank(MRR@N): evaluates a recommender system that predicts a ranked list of items. N denote the number of items that are at the top of the recommendation list to be evaluated. For example, MRR@10 evaluates a recommended list containing the Top 10 items. It is defined by:

$$MRR@N = \frac{1}{\mid C'_u \mid} \sum_{u \in U} RR(u)$$

$$RR(u) = \sum_{c \in C'_u} \frac{1}{rank_u(d)} \tag{18}$$

where C'_u is the enrollment set of student u in the test set and $rank_u(d)$ is the rank of course c in the Top-N recommendation result for student u, which is R_u.

ii) Recall@N: is the proportion of relevant items in the Top-N recommendation list.

$$RECALL@N = \sum_{u \in U} \frac{1}{\mid U \mid} \frac{\sum_{c \in R_u} (c \in C'_u)}{min(N, \mid C'_u \mid)} \tag{19}$$

iii) Precision@N (Prec@N): the fraction of relevant items in the top k recommendations

$$PREC@N = \sum_{u \in U} \frac{1}{\mid U \mid} \frac{\sum_{c \in R_u} (c \in C'_u)}{min(N, \mid R_u \mid)} \tag{20}$$

iv) F1 Score (F1@N): combines recall and precision in a single number evaluation metric. This makes the comparison across different models much more straightforward.

$$F1@N = 2 * \frac{Prec@10 * Recall@10}{Prec@10 + Recall@10} \tag{21}$$

3) Parameter settings

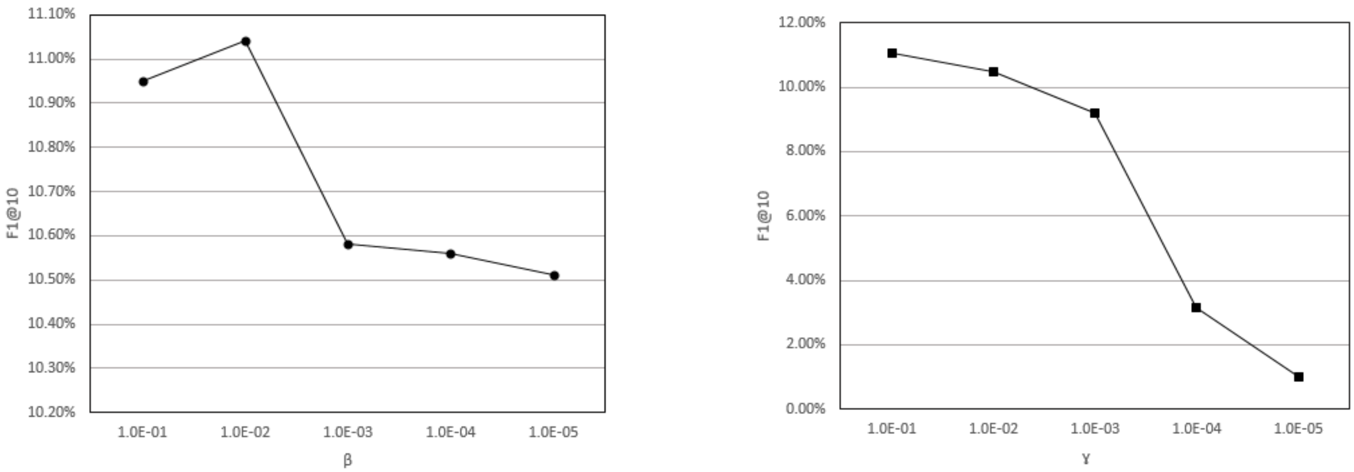
For comparison, we determined the ideal parameters for each technique either by following the recommendations in the related articles or by conducting a grid search in our experiments. For the bipartite graph-based methods, BiNE [17] and our proposed model, PreBiGE, the parameters searched include γ and β since they play an important role in balancing the impact of the high-order relations and the prerequisite relations, as well as the direct relations. β is searched from $[0.0001, 0.0001, 0.001, 0.01, 0.1]$, γ is also searched from γ $[0.0001, 0.0001, 0.001, 0.01, 0.1]$. The result is shown in Figure 3a and 3b. The learning rate and regularizer were also searched. The learning rate λ is searched from $[0.01, 0.025, 0.05, 0.01, 0.1]$ and the regularizer δ is searched from $[0.01.0.025, 0.01, 0.1]$

The dimension of the latent components is set to 128 for all techniques. Based on the grid search the optimal setting of β and γ as seen in Figure 3a and 3b is set to 0.01 and 0.1, respectively. This setting applies to BiNE [17] and our proposed model. The learning rate and regulariser are also set to 0.1 and 0.1, respectively for BiNE [17] and our proposed model, PreBiGE. The dropout rate, learning rate and regularization coefficient is set to 10^{-1}, 10^{-3} and 10^{-4}, respectively for NGCF [18] and LightGCN [28]. For NeuMF and BPR, the learning rate is set to 10^{-2}.

C. Performance Comparison (RQI)

We compare the performance of our model with existing recommendation methods. Table III shows the performance comparison. We have the following observations:

The result of the best baseline on each metric is marked with † while our model, PreBiGE, is marked with $*$. The last row is the percentage improvement of PreBiGE compared to the best baseline, defined as $\frac{PreBiGE-baseline}{baseline}$. We make several observations on the baselines. We can see that the models, BiNE, NGCF and LightGCN, utilizing the high-order collaborative relations perform better compared to the models, NeuMF and BPR, utilizing only the direct relation



(a) F1@10 Vs β

(b) F1@10 Vs γ

Fig. 3. Parameters β and γ analysis

TABLE III. Performance comparison of PreBiGE with existing models (in %).

Models	**Prec@10**	**Recall@10**	**F1@10**	**Mrr@10**
BPR [26]	4.09	17.64	6.64	14.94
NMF [27]	4.42	18.91	7.16	15.26
BiNE [17]	5.76	24.89	9.36	17.83
NGCF [18]	6.49	27.29	10.48	19.21
LightGCN [28]	†6.56	† 28.22	†10.64	†19.23
PreBiGE	*6.81	*29.18	*11.04	*19.50
Improv.	3.68	3.29	3.61	1.38

'*' and 'Improv.' denote the statistical significance at $p < 0.01$ with a paired t-test and improvement of our model compared to the best method (†), respectively

between entities. Furthermore, PreBiGE utilizing the high-order relations, prerequisite dependency and direct relations, outperforms baselines, BiNE, NGCF and LightGCN, utilizing high-order relations and direct relation only. Table III also shows that our model, PreBiGE, outperforms all baselines. This shows the benefit of utilizing the prerequisite relation between courses. In summary, our model outperforms all baselines which answer the research question (RQI).

D. Influence of Course Prerequisite Dependency (RQ II)

In this section, we analyze how the prerequisite relations between courses influence the performance of the recommendation model. Table IV shows the performance of our model, $PreBiGE$, utilizing the prerequisite dependency and its variant, $PreBiGE_{WOP}$, that does not utilize the prerequisite dependency between courses extracted from the knowledge graph. It can be seen that the model utilizing the prerequisite dependency significantly outperforms the other model. This indicates that capturing the prerequisite dependency greatly influence the performance of recommendation result answering our research question (RQII).

TABLE IV. Influence of Course Prerequisite Dependency (in %).

Models	**Prec@10**	**Recall@10**	**F1@10**	**Mrr@10**
$PreBiGE_{WOP}$	6.54	27.75	10.58	18.52
$PreBiGE$	6.81	29.18	11.04	19.50
Improv.	3.99	4.90	4.17	5.03

E. Influence of Weighted Course Prerequisite Graph (RQ III)

In this section, we analyze whether giving different strengths to the prerequisite relation between course pair influence the recommendation result. To demonstrate the effectiveness of giving different strengths to the prerequisite relations between pairs, we compare our model, $PreBiGE$, whose course prerequisite graph contains edge weights representing the number of paths between the pairs and another variant, $PreBiGE_{UW}$, in which the prerequisite graph is an unweighted graph in which the relations between all course pairs are equal. The number of paths between course pairs represented as weights in the course prerequisite graph differentiates the strength of the prerequisite relations. Table V shows the performance between these two variants. It can be seen that our model, PreBiGE, utilizing a weighted prerequisite relation significantly outperforms the other variant which is unweighted. This indicates that assigning different weights between course pairs in the prerequisite graph influence the performance of recommendation result answering our research question (RQIII).

TABLE V. Influence of Weighted Course Prerequisite Graph (in %).

Models	**Prec@10**	**Recall@10**	**F1@10**	**Mrr@10**
$PreBiGE_{UW}$	6.56	28.27	10.65	19.29
$PreBiGE$	6.81	29.18	11.04	19.50
Improv.	3.63	3.12	3.53	1.07

F. Parameters sensitivity analysis (RQIV)

In this section, we investigate the influence of hyperparameters β and γ (we fix $\alpha = 0.1$). These two parameters play a crucial role in balancing the prerequisite relation, the high-order relations and the direct relations between nodes in the graph. Except for the parameters being tested, we assume default values for other hyperparameters. From Figure 3a, we observe that with the decrease in β, the performance increases and then drastically starts to decrease showing the highest performance at 0.01. From Figure 3b, we observe that with the decrease in γ the performance is decreasing significantly showing highest performance at 0.1. We report the result of our model with these two settings.

VI. Conclusion

We have presented a novel model for recommending courses using course prerequisite relation graph and student-course bipartite graph embedding. Our model, PreBiGE, jointly utilizes the direct relation between students and courses, the high-order collaborative relation between sets of students and sets of courses in the bipartite graph, as well as the prerequisite relation between the set of courses in the course prerequisite graph. Our model augments the high-order collaborative relations between courses with their prerequisite relations. Experiments on real-world dataset show that it significantly improves the recommendation result compared with other baseline methods.

In future study, we will consider the chronological order by which each student interacts with the content of a course to understand the users' behaviour for better personalization.

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.

Data Availability Statement

The dataset used in this study is openly available in the Large-scale Data Repository for NLP Applications in MOOCs (MOOCCube) at http://moocdata.cn/data/MOOCCube.

References

[1] L. Liu, F. Lin, B. Wang, K. Li, M. Xiao, J. Xiahou, and P. Wu, "Prerequisite Relation Learning for Course Concepts Based on Hyperbolic Deep Representation," *IEEE Access*, vol. 8, pp. 49 079–49 089, 2020.

[2] Z. Zhao, Y. Yang, C. Li, and L. Nie, "GuessUNeed: Recommending Courses via Neural Attention Network and Course Prerequisite Relation Embeddings," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 16, no. 4, 2021.

[3] P. Symeonidis and D. Malakoudis, "Multi-modal matrix factorization with side information for recommending massive open online courses," *Expert Systems with Applications*, vol. 118, pp. 261–271, 2019. [Online]. Available: https://doi.org/10.1016/j.eswa.2018.09.053

[4] J. Li, C. Chang, Z. Yang, H. Fu, and Y. Tang, *Probability Matrix Factorization Algorithm for Course Recommendation System Fusing the Influence of Nearest Neighbor Users Based on Cloud Model*. Springer International Publishing, 2019, vol. 11354 LNCS. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-15127-0_49

[5] H. Zhang, T. Huang, Z. Lv, S. Liu, and H. Yang, "MOOCRC: A Highly Accurate Resource Recommendation Model for Use in MOOC Environments," *Mobile Networks and Applications*, vol. 24, no. 1, pp. 34–46, 2019.

[6] H. K. Ahmad, C. Qi, Z. Wu, and B. A. Muhammad, "ABiNE-CRS : course recommender system in online education using attributed bipartite network embedding," *Applied Intelligence*, 2022.

[7] X. Yang and W. Jiang, "Dynamic Online Course Recommendation Based on Course Network and User Network," in *7th International Conference of Smart City and Informatization , iSCI*, 2019, pp. 180–196.

[8] J. Wang, H. Xie, F. L. Wang, L.-K. Lee, and O. T. S. Au, "Top-N personalized recommendation with graph neural networks in MOOCs," *Computers and Education: Artificial Intelligence*, vol. 2, no. September 2020, p. 100010, 2021. [Online]. Available: https://doi.org/10.1016/j.caeai.2021.100010

[9] A. da S. Dias and L. K. Wives, "Recommender system for learning objects based in the fusion of social signals, interests, and preferences of learner users in ubiquitous e-learning systems," *Personal and Ubiquitous Computing*, vol. 23, no. 2, pp. 249–268, 2019.

[10] X. Jing and J. Tang, "Guess you like: Course recommendation in MOOCs," in *Proceedings - EEE/WIC/ACM International Conference on Web Intelligence*, 2017, pp. 783–789.

[11] B. Yang, Y. Lei, J. Liu, and W. Li, "Social Collaborative Filtering by Trust," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, pp. 1633–1647, 2017.

[12] S. Roy, M. Madhyastha, S. Lawrence, and V. Rajan, "Inferring Concept Prerequisite Relations from Online Educational Resources," in *AAAI Conference on Innovative Applications of Artificial Intelligence*, 2019, pp. 9589–9594.

[13] L. Pan, C. Li, J. Li, and J. Tang, "Prerequisite Relation Learning for Concepts in MOOCs," in *55th Annual Meeting ofthe Association for Computational Linguistics*, 2017, pp. 1447–1456.

[14] C. Liang, J. Ye, S. Wang, B. Pursel, and C. L. Giles, "Investigating Active Learning for Concept Prerequisite Learning," in *8th AAAI Symposium on Educational Advances in Artificial Intelligence 2018 (EAAI-18)*, vol. 2018, 2018, pp. 7913–7919.

[15] J. Yu, G. Luo, T. Xiao, Q. Zhong, Y. Wang, W. Feng, J. Luo, C. Wang, L. Hou, J. Li, Z. Liu, and J. Tang, "MOOCCube: A Large-scale Data Repository for NLP Applications in MOOCs," in *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, 2020, pp. 3135–3142.

[16] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *WWW 2015 - Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.

[17] M. Gao, X. He, L. Chen, T. Liu, J. Zhang, and A. Zhou, "Learning Vertex Representations for Bipartite Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 379–393, 2022.

[18] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 165–174.

[19] B. Perozzi and S. Skiena, "DeepWalk: Online Learning of Social Representations," in *Proceedingds ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD*, 2014, pp. 701–710.

[20] H. Deng, M. R. Lyu, and I. King, "A generalized Co-HITS algorithm and its application to bipartite graphs," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 239–247.

[21] A. Grover and J. Leskovec, "Node2Vec," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD*, 2016, pp. 855–864.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems NIPS*, 2013, pp. 3111–3119.

[23] X. Amatriain and J. M. Pujol, *Data Mining Methods for Recommender Systems*. Boston, MA: Springer US, 2015, pp. 227–262. [Online]. Available: https://doi.org/10.1007/978-1-4899-7637-6_7

[24] I. Uddin, A. S. Imran, K. Muhammad, N. Fayyaz, and M. Sajjad, "A Systematic Mapping Review on MOOC Recommender Systems," *IEEE Access*, vol. 9, pp. 118 379–118 405, 2021.

[25] K. Verbert, H. Drachsler, N. Manouselis, and M. Wolpers, "Dataset-driven research for improving recommender systems for learning Dataset-driven Research for Improving Recommender Systems for Learning," in *1st International Conference on Learning Analytics and Knowledge*, no. February, 2011, pp. 44–53.

[26] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009*, 2009, pp. 452–461.

[27] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW '17. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017, p. 173–182.

[28] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation," in *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020. [Online]. Available: http://arxiv.org/abs/2002.02126

Hafsa Kabir Ahmad received her B.Sc. and M.Sc. degrees in computer science from Bayero University, Kano, in 2011 and 2015, respectively. She is currently pursuing a PhD degree in computer science at Shaanxi Normal University, China. She is also a Lecturer with the Department of Computer Science, Bayero University, Kano, Nigeria. Her current research interests include recommender systems and network representation learning.

Bo Liu received his M.S. degree from Xi'an Jiaotong University, Xi'an, China and Ph.D. degree in the School of Systems Information Science, Future University Hakodate, Hokkaido, Japan and is also a faculty member in the School of Computer Science, Shaanxi Normal University, Xi'an, China. His current research interests include Nanocommunications and Nano-networks.

Bello Ahmad Muhammad received his B.Sc. and M.Sc. degrees in computer science from Bayero University, Kano, Nigeria, in 2010 and 2015, respectively. He is currently pursuing a PhD degree in Computer Science at Shaanxi Normal University, China. He is also working with the University Library, Bayero University, Kano, Nigeria. His research interests include Learning style detection, recommender system, and graph representation learning.

Mubarak Umar received his PhD in Computer Science at Shaanxi Normal University, China. He is also a Lecturer with the Department of Information Technology, Bayero University, Kano, Nigeria. His research interests include authentication in body area networks, sensor networks security, information security of wireless communication, and cryptography.