International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 8241-8255

https://doi.org/10.70517/ijhsa464705

Building and maintaining knowledge graphs: Application of deep learning methods for course knowledge recommendation

Jicang Xu[1,*]
[1] School of Economics and Management, China University of Petroleum, Beijing, 102249, China
Corresponding authors: (e-mail: xujicang2024@126.com).

Abstract With the rapid development of technologies such as big data and artificial intelligence, the next generation of artificial intelligence technologies centered on knowledge graphs has gradually matured, making it possible for teaching knowledge graphs to assist teachers in achieving smart classroom teaching. Based on an introduction to knowledge graph-related technologies and deep learning theories, this paper takes course knowledge as the research object, constructs a course knowledge ontology model and knowledge storage process. Using the constructed course knowledge graph as the foundation, the paper combines stacked LSTM and GCN models to build a course knowledge recommendation model. Stacked LSTM is used for entity-relation extraction, and GCN is used for knowledge mapping, thereby enhancing the effectiveness of course knowledge recommendations. The study found that when the TopN recommendation value was set to 10, the average accuracy of the proposed method was 0.308, which was 1.18 times higher than the UserCF method. Additionally, the knowledge graph, stacked LSTM, and GCN in the model all had a significant impact on improving the performance of course knowledge recommendation. By leveraging deep learning technology, knowledge entities in the data can be better distinguished, thereby facilitating the establishment of a knowledge ontology model and providing a solid foundation for knowledge mapping and recommendation.

Index Terms knowledge graph, ontology model, deep learning, stacked LSTM, GCN model

I. Introduction

Knowledge is processed information and also the foundation of artificial intelligence [1]. Human demand for knowledge is primarily expressed through natural language, and question-answering systems centered on questions are currently a hot research topic in the field of natural language processing [2], [3]. These systems allow users to ask questions in natural language and provide concise, relatively accurate answers [4]. Knowledge representation involves using a set of symbols to represent knowledge in a structure that computers can process [5]. The primary methods of representation include first-order predicate systems, production rule systems, semantic networks, and frame theory. Ontologies, which are semantic information models, are the primary modeling tools for describing semantic knowledge within semantic networks [6]. The construction of domain ontology repositories not only clearly describes the concepts and relationships between industry knowledge but also enables the sharing and reuse of industry knowledge, facilitating the management and maintenance of industry knowledge repositories. Additionally, it plays a crucial role in answering questions within the domain [7], [8].

Internationally, since English does not require word segmentation and has relatively complete grammatical rules, the ambiguity of English sentences is significantly reduced compared to Chinese sentences. The analysis of questions is primarily conducted through dependency parsing trees [9], [10]. Most researchers have focused their efforts on ontology construction and reasoning, employing rule-based methods such as JESS and SWRL, which are well-suited for expert systems and recommendation systems [11]-[13]. However, these methods are highly domain-dependent, and heuristic rules are often fragile. Additionally, the cost of manually constructing ontologies is extremely high. To address this, Sicilia, Á, and others introduced a web-based visual ontology mapping editor (Map-On), which simplifies the mapping creation process through a graphical interface and the automatic generation of R2RML documents [14]. Iglesias-Molina, A et al. proposed a conceptual mapping semantic model aimed at reflecting the characteristics and features of existing declarative mapping languages in the process of constructing knowledge graphs [15]. Gould, N and Mackaness, W utilized ontology modeling methods to achieve automated generation of design processes, with the focus on integrating key information and background information through a mapping of generalized knowledge and geographical features [16]. Zhao, M et al. proposed a method called FCA-Map, which utilizes formal concept analysis to identify and validate mapping relationships between different semantic models, demonstrating its effectiveness and competitiveness in large-scale biomedical semantic model matching tasks [17].

Most research methods for Chinese questions based on domain-specific ontologies treat the constructed domain ontology as a knowledge repository for storing knowledge, while question processing is primarily divided into two categories: syntactic analysis of questions, semantic similarity calculation, or matching them with sentence templates [18]-[20]. Liu, J et al. proposed a new ontology learning model named Domain Ontology Graph (DOG), which defines ontologies and knowledge conceptualization through domain-specific textual documents and provides a semi-automatic learning method. In simulated studies on Chinese text data, it demonstrated high classification accuracy [21]. Yin, P et al. proposed an ontology-based semantic model for identifying features and opinions in Chinese review texts, which enhances opinion mining and sentiment analysis by converting unstructured reviews into structured data [22]. Li, Z et al. developed a domain ontology model for preserving and disseminating heritage information related to traditional Chinese culture using a combination of top-down and bottom-up approaches. This ontology meets quality standards and aids in organizing traditional Chinese culture and constructing related databases [23]. Liu, J et al. proposed a fuzzy set-based knowledge ontology representation method and an ontology mapping algorithm combining similarity calculation and support vector machines (SVM), which can expand the application of knowledge ontologies and improve information retrieval capabilities in the semantic web [24]. Xie, K et al. proposed a domain-independent ontology learning method (TF-Mnt) that utilizes transfer learning to efficiently transfer existing domain knowledge to new domains under conditions of limited labeled data, addressing the challenges of reducing labeling workloads and achieving good performance on real-world datasets [25].

This paper introduces knowledge graph construction techniques, discusses the feasibility of the Neo4j open-source database for knowledge graph storage, and provides a detailed explanation of the specific principles of graph convolutional neural networks and long short-term memory neural networks in deep learning. MOOC online teaching courses are selected as the research object, and their course knowledge ontology model and knowledge storage process are constructed. By combining knowledge graphs with stacked LSTM and GCN models, a course knowledge recommendation model is constructed. In this model, the stacked LSTM model is used for course knowledge entity relationship extraction, and the GCN model is introduced to achieve course knowledge mapping, thereby enabling efficient course knowledge recommendation.

II. Related theories and technologies

In recent years, the state has strongly supported the smart education industry and artificial intelligence industry, promoting the development of educational informatization. Various smart education courses and platforms have entered the public eye, bringing convenience to users while also raising an issue. Specifically, for courses with a large volume of knowledge points, particularly those like data structures that are highly logical, contain many abstract concepts, and are closely intertwined with other specialized courses, learners often encounter difficulties in understanding abstract concepts and grasping the relationships between knowledge points during self-study. Therefore, it is necessary to organize the knowledge and relationships between knowledge points within data structure courses.

II. A.Knowledge Graph Construction Technology

II. A. 1) Knowledge Graph Construction

A knowledge graph is a type of graph data structure composed of nodes and edges, which includes entities, relationships, and attributes. Nodes correspond to “entities,” edges correspond to “relationships,” and ‘attributes’ are embedded in nodes and edges, thereby forming a triplet representation of “head entity, relationship, tail entity.” This concept originated from the need to structure and link information, with the aim of more intuitively expressing and understanding knowledge in the real world. Entities refer to specific objects, concepts, or events in the real world, such as people, things, buildings, countries, etc. A relationship describes the association or interaction between one entity and another, defining the semantic connections between different entities, such as “contains,” “owns,” or “is located in.” An attribute refers to the characteristics or properties associated with an entity or relationship. For example, attributes of a person as an entity may include name, address, occupation, etc. Attributes typically exist in key-value pairs and can be represented using the triplet “<entity, attribute, value>.”

Figure 1 shows the knowledge graph construction process. There are two mainstream methods for constructing knowledge graphs. The first is a bottom-up approach, which focuses on extracting information from raw data to gradually construct the knowledge graph. This involves analyzing and processing large amounts of structured or unstructured data to identify and extract entities from the data, analyze the relationships within the data, establish connections between entities, and build an ontology based on the extracted information to define the structure of entities, attributes, and relationships. The second is the top-down approach, which emphasizes defining the ontology and model first, designing entities and their relationships based on the ontology, and then constructing the knowledge graph by filling in instance data. The construction of a knowledge graph is a complex process that includes key steps such as knowledge acquisition, knowledge representation, knowledge extraction, knowledge fusion, and knowledge storage [26].

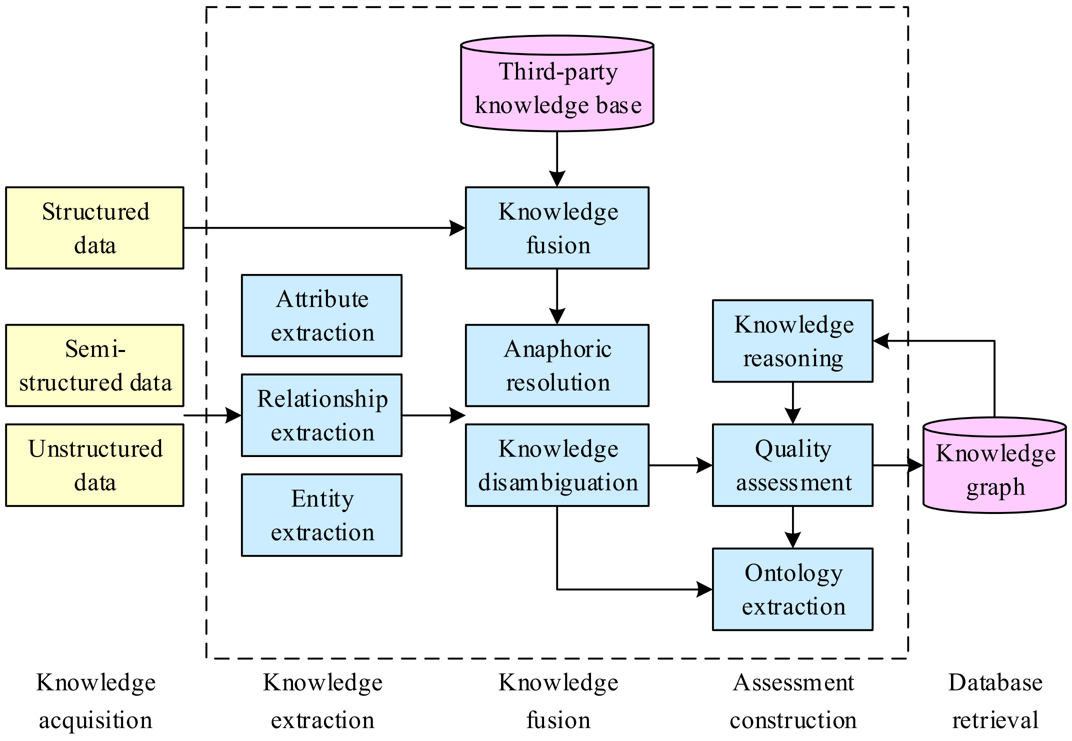


Figure 1: The process of knowledge graph construction

II. A. 2) Neo4j Open Source Database

Most databases offer good documentation quality and portability, with some databases providing free use but with node restrictions. Property graph functionality is widely supported, while only Trinity supports hypergraphs, and AllegroGraph uses the RDF format. The graph database Neo4j offers excellent documentation quality, high portability, and a completely free usage model, making it suitable for a wide range of development and deployment needs. Therefore, this paper selects Neo4j as the database for knowledge graph data storage and analysis.

Neo4j's storage model is based on graph theory structures, primarily storing three elements: nodes, relationships, and properties. Nodes and relationships each have unique identifiers, nodes are connected via relationships, and relationships can also carry property information. All node and relationship properties are stored as key-value pairs. Additionally, Neo4j uses index structures such as B-trees and R-trees to accelerate data retrieval and employs linked list structures to achieve efficient storage and retrieval.

In the field of course knowledge recommendation, Neo4j can store all course data, making it suitable for handling complex course knowledge association data, such as multiple relationships between courses, students, and learning outcomes. Through an efficient query mechanism, it helps students or teachers quickly find course knowledge-related information and generate personalized learning recommendations.

II. B.Deep Learning-Related Theories

II. B. 1) Graph Convolutional Neural Networks

Graph Convolutional Neural Networks (GCNs) are a type of graph neural network model. GCNs can aggregate node features and neighboring node features through convolution operations, effectively capturing local graph structural information while maintaining high computational efficiency. In principle, GCNs aggregate information from each node's neighboring nodes and feed it into the neural network. The propagation formula for GCNs is:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{1}$$

where $\tilde{A} = A + I_N$, I_N is an n-dimensional identity matrix. Since the diagonal elements of the adjacency matrix A are all 0, this causes the node's own information to be ignored when multiplying by the feature matrix H for information aggregation. This operation allows the node's own features to be included in the calculation. $\tilde{D}$ is the degree matrix of $\tilde{A}$, i.e., the number of nodes connected to each node, and its structure is a diagonal matrix. To prevent the feature aggregation process from overly favoring nodes with more edges, $\tilde{A}$ needs to be normalized to obtain the normalized symmetric matrix $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$. $H^{(l)}$ denotes the activation function of the l th layer in the neural network, and $W^{(l)}$ denotes the feature matrix.

Convolutional neural networks can be stacked in multiple layers. When it has l hidden layers, it can be represented as follows:

Input layer: $h_v^{(0)} = x_v$.

First hidden layer: $h_v^{(1)} = \mathrm{Re}LU\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}h_v^{(0)}W^{(0)}\right)$.

l th hidden layer: $h_v^{(l)} = \mathrm{Re}LU\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}h_v^{(l-1)}W^{(l-1)}\right)$.

Here, we assume that the nonlinear activation function is $\mathrm{Re}LU$. After l hidden layers of computation, we obtain the feature vector $h_v^{(l)}$, which contains the required information. After operating on it according to the task requirements, we can achieve different task objectives [27].

II. B. 2) Long Short-Term Memory Network

Long Short-Term Memory (LSTM) neural networks are a derivative of Recurrent Neural Networks (RNNs), overcoming the shortcomings of RNNs and effectively capturing semantic relationships between long sequences. To address the issue of gradient vanishing, LSTM structures are more complex than those of RNNs. The LSTM structure primarily consists of four components: the forget gate, the input gate, the cell state, and the output gate. These components work together to enable LSTM to effectively process and remember long sequence data [28].

(1) Forget gate. The forget gate controls which information in the cell state should be discarded or forgotten, helping the model remember long-term dependencies while ignoring information irrelevant to the current task. The forget gate acts on the cell state Ct-1, controlling the retention and forgetting of information and selectively retaining useful information. The calculation formula is as follows:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2}$$

Two vectors of the same dimension, W_f and h_{t-1}, and W_f and x_t, are multiplied, then added to the bias b_f, and finally fed into the sigmoid function to obtain the matrix f_t. Each value in the matrix determines the retention of the information corresponding to C_{t-1}.

(2) Input gate. The input gate determines which information the cell state can store, enabling selective memory of new information. The process of remembering new information can be divided into two parts. The first part involves constructing the input gate to determine the content of the new memory. The input gate performs a linear operation to obtain a matrix i_t of the same dimension as C_t, where each value in the matrix determines the information retained by C_t. The second part involves constructing a candidate cell state $\tilde{C}_t$ that preserves the information from x_t and h_{t-1}. Multiplying this by the values of i_t determines which information to retain, thereby using the retained information as the newly added cell state. The calculation formula is as follows:

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{3}$$

$$\tilde{C}_t = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{4}$$

(3) Update gate. The forget gate and input gate determine which information to discard and which to remember, thereby updating the cell. There are two parts to updating the new cell state: one is the memory left behind after the

previous cell discards the discarded information; the other is the input gate filtering the information and then using the filtered useful information as new memory. The formula for updating the new cell state is as follows:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{5}$$

In the equation, f_t multiplied by C_{t-1} represents the processed old memory, and i_t multiplied by $\tilde{C}_t$ represents the new memory to be added. Together, they represent the new cell state.

(4) Output gate. The output gate determines the final output content of h_t based on the cell state C_t. The first step uses a sigmoid layer to calculate O_t to determine the portion of the cell state that needs to be output. The second step processes the new cell state using the tanh function, mapping its value range to $[-1,1]$, and then performs element-wise multiplication with O_t to control the output content. The formula is as follows:

$$o_t = \sigma\left(W_o[h_{t-1}, x_t] + b_o\right) \tag{6}$$

$$h_t = o_t \cdot \tanh \cdot C_t \tag{7}$$

III. Recommendation model based on knowledge graphs

With the widespread adoption of the “Internet+Education” model, the education sector has accumulated a vast amount of course resources with practical application value. However, course resources from different sources have varying structures and lack established connections, making it difficult to utilize them effectively in teaching. Additionally, existing textbooks, learning websites, and other methods of organizing course knowledge are limited in their approach, primarily focusing on text-based content and rarely integrating course resources across different modalities to achieve greater effectiveness. There is an urgent need for emerging technologies such as artificial intelligence and big data to collect and process relevant course resources, providing intelligent services for teachers and students, and continuously promoting the intelligent development of educational applications.

III. A. Data Acquisition and Ontology Model Construction

III. A. 1) Course Knowledge Data Acquisition

In the process of constructing a course knowledge graph, the data collection and preprocessing stage is a critical foundational step. This study employs a multi-source heterogeneous data collection strategy, leveraging multiple channels such as textbooks and online course resources to ensure the comprehensiveness and authority of the data. In this process, the MOOCCubeX dataset demonstrates its unique value in constructing course knowledge graphs. MOOCCubeX is a large-scale, multimodal MOOC (Massive Open Online Course) dataset encompassing rich course content, learning behavior, and assessment data. The dataset not only covers core course knowledge points but also includes a large number of exercises, video explanations, and learner interaction information, providing an ideal data foundation for constructing a comprehensive and dynamic knowledge graph.

MOOCCubeX is a large-scale MOOC course dataset maintained by the Knowledge Engineering Laboratory at Tsinghua University. The dataset includes courses, videos, exercises, fine-grained concepts, and over 100 million student learning behavior data points from the XuetangX online platform. MOOCCubeX organizes heterogeneous data using a fine-grained concept graph, making resources more precise, easier to express, retrieve, and model. MOOCCubeX encompasses rich course resource data, student behavior data, and concept link data, providing a high-quality data foundation for course knowledge graph construction.

III. A. 2) Course Knowledge Ontology Model

The knowledge entity model layer provided by ontology construction is the core of the top-down approach to building knowledge graphs. It formally represents entities, relationships, and attributes using ontology languages such as OWL and RDF, enabling ontology reasoning through ontology tools to achieve the representation, sharing, and utilization of knowledge. The ontology construction process consists of seven steps: requirement analysis, concept definition, relationship definition, model construction, data maintenance, validation and evaluation, and application and update. This study follows the seven-step ontology construction method to build a course knowledge ontology model. Based on the core concept set explicitly outlined in the new curriculum standards, the top-level design of the knowledge ontology is designed, knowledge entities are encapsulated, entity relationships are constrained, and the ontology is constructed and stored using the Protégé tool.

Figure 2 shows the framework of the course knowledge ontology model. Learners assess their knowledge mastery by completing exercises at their current ability level and organize knowledge points as they advance to higher ability levels. If they fail to pass the exercises at a certain level, the corresponding learning resources are displayed to assist learners in completing online self-study. If learners still fail the exercises after using the learning resources or have questions about the knowledge points in that section, the teacher will explain them again during the lesson.

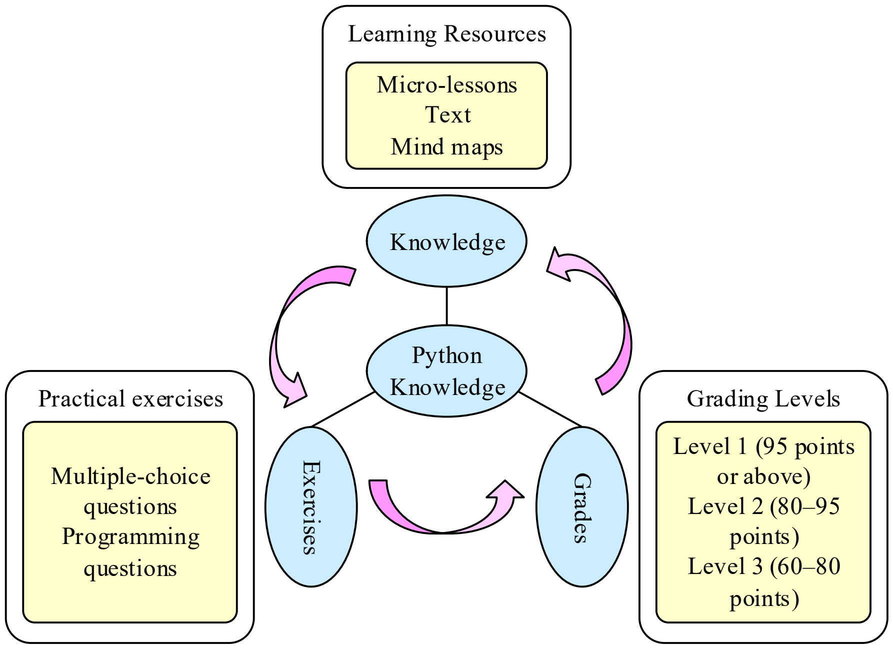


Figure 2: Curriculum knowledge ontology model framework

III. A. 3) Course Knowledge Storage Process

Based on the course knowledge graph construction process, this paper uses the course knowledge ontology model and Neo4j graph database to store knowledge data other than course topics, and uses the Cypher language for visual queries to complete the construction of the course knowledge graph. To facilitate the storage of course knowledge data, this paper adopts the CSV format import method for storage. The data is read using the LOAD CSV command in the Cypher language. This method is commonly used for storage scenarios with approximately 100,000 nodes. Its characteristics include flexible usage and the ability to modify data in real time. The knowledge storage process is as follows:

Step 1: Save the data in CSV format. First, generate a separate CSV file for each entity data. Note that the format of Neo4j graph data is UTF-8, so the CSV files for each entity data in the course knowledge must also be stored in UTF-8 format. Then, locate the folder named “import” in Neo4j and place the CSV files for each entity data into it.

Step 2: Load entity information.

Step 3: Establish relationships.

Step 4: Complete storage.

After completing the above steps, the storage of course knowledge entities and their relationships is completed. After visualizing the query, the course knowledge graph is obtained.

III. B. Design of a Recommendation Model for Course Knowledge

III. B. 1) Knowledge Recommendation Model Framework

The course knowledge recommendation model (KG-SLSTM-GCN) framework based on knowledge graphs is shown in Figure 3. A knowledge graph is constructed using course content information and user behavior records to provide semantic relationships between courses. Next, user interaction history data is serialized, and stacked LSTM is used to process each user interaction, effectively capturing the temporal characteristics of user behavior and generating user feature vectors. These vectors are weighted and fused with user feature vectors generated by the GCN model to enhance user feature representation. Combined with the graph convolutional network, the model deeply explores the relationships between courses and predicts user ratings for courses. The prediction results can be expressed as:

$$\hat{y}_{uv} = F(u, v \mid \Theta, Y, G) \tag{8}$$

In the equation, Θ represents the model parameters, Y represents the user-item interaction matrix, and G represents the knowledge graph. $\hat{y}_{uv}=1$ if and only if user u interacts with course v. If there is no interaction, then $\hat{y}_{uv}=0$.

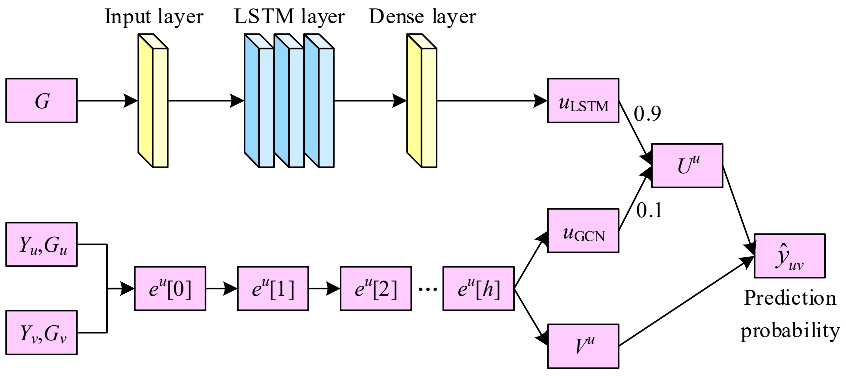


Figure 3: Curriculum knowledge recommendation model

The model process is as follows:

(1) Input the user-project interaction matrix Y and knowledge graph G.

(2) Iterate through the user-item pairs (u,v) in the interaction matrix Y and calculate the receptive field of course v.

(3) Calculate the neighborhood representation $v^u_{x(v)}$ of course v and aggregate it with its own representation.

(4) Repeat step (3) H times to obtain the final H-order representation V^u.

(5) Fusion the user feature vector u_{LSTM} generated by the stacked LSTM with the user feature vector u_{GCN} generated by the GCN to obtain the final user feature vector U^u.

(6) Input U^u and V^u into the prediction function $\hat{y}_{uv}$.

III. B. 2) Entity Relationship Extraction Module

The entity or attribute extraction in this paper primarily targets unstructured data, as the dataset is too large and the information contained in each individual dataset is relatively sparse. Traditional processing methods are not suitable for such data. Based on the characteristics of the data in this paper, the BIEO method is adopted for entity annotation, which involves marking each element in the input text sequence without pre-segmentation.

Based on an analysis of text features and their expressive characteristics, this paper employs a stacked LSTM model to extract bidirectional text semantic information. The output text latent vectors not only include normal forward-narrative text (left-to-right) but also output reverse backward-narrative text information (right-to-left). Based on the pre-trained BERT model using Wikipedia as a foundation, professional text data from the database—i.e., a large number of course books—is imported to convert complex static vectors into dynamic vectors tailored to the contextual environment of words, thereby enhancing word position information and reducing issues such as polysemy and ambiguous semantics.

After converting the data into vector representations, it undergoes dual stacking processing via forward LSTM and backward LSTM, yielding implicit information containing bidirectional semantic meaning in the text. In the CRF layer, the probabilities of various possibilities for the target label in the sequence are calculated, and the resulting probability matrix can be expressed as:

$$S(x,y)=\sum_{i=0}^{n}A_{y_i,y_{i+1}}+\sum_{i=1}^{n}P_{i,y_i} \tag{9}$$

where n denotes the text length, A denotes the probability of transitioning from the current state label to all labels, and P denotes the probability that the current element belongs to the label state in the data. The transition matrix for label transition probabilities is generally obtained through the loss function:

$$\log\left(P(y\mid x)\right)=S\left(x,y\right)-\log\sum_{\bar{y}\in \mathrm{Y}_x}s\left(x,\bar{y}\right) \tag{10}$$

The optimal label path is obtained using the Viterbi algorithm. That is:

$$y^* = \arg\max S\left(x, \overline{y}\right) \tag{11}$$

III. B. 3) Course Knowledge Mapping Module

To more effectively recommend course knowledge, this paper proposes a course knowledge mapping module based on graph convolutional neural networks, which can extract and retain all course detail feature information as much as possible, ensuring the acquisition of clear course knowledge semantic information.

Specifically, given an input feature $F \in R^{(B\times C\times H\times W)}$, this paper uses global average pooling to obtain the global feature $F_g \in R^{(B\times C\times 1\times 1)}$ for each channel. The global feature F_g is then converted into a graph feature $X \in R^{(1\times b\times 1\times C)}$ and fed into the graph convolutional layer. The graph convolutional layer can then use the defined aggregator to collect information from neighboring nodes and uncover the relationships between different nodes. That is:

$$GraphConv(X) = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\Theta \tag{12}$$

In this context, $\tilde{D}$ represents the degree matrix, $\tilde{A}$ represents the adjacency matrix, and $\Theta \in R^{C\times F}$ represents the convolution kernel parameters for graph convolution.

The output graph features are then restored to a global feature map $F_g' \in R^{(B\times C\times 1\times 1)}$, where F_g' contains inter-channel correlation information. This is multiplied by the original input features, effectively performing a weighted operation on the original features, emphasizing important features and weakening unimportant ones. A 1×1 convolution is then used to reduce the dimension of the feature map and fuse the features, reducing the dimension of unimportant features while retaining as much detail as possible. Finally, the mapped feature $F_o \in R^{(B\times C\times H\times W)}$ is obtained.

III. B. 4) Course Knowledge Recommendation Module

To accomplish the recommendation task, graph neural networks are used to perform representation learning on the above graph structures. By comparing and optimizing the representation results of different views, high-quality user and project representations are obtained.

GCN is used to learn node representations in the user-project bipartite graph J, the user-entity subgraph G', and the project-entity subgraph g. The training process for the user-project bipartite graph J is as follows:

$$e_u^{(k+1)} = \sum_{i\in N_u} \frac{1}{\sqrt{|N_u||N_i|}} e_i^{(k)} \tag{13}$$

$$e_i^{(k+1)} = \sum_{u\in N_i} \frac{1}{\sqrt{|N_u||N_i|}} e_u^{(k)} \tag{14}$$

Among these, N_u denotes the items that interact with user u, and similarly, N_i denotes the users who have interacted with item i. $e_u^{(k+1)}$ denotes the representation of user u learned by GCN at the $(k+1)$th layer, and $e_i^{(k+1)}$ denotes the representation of item i learned by GCN at the $(k+1)$th layer. By summing the results from multiple layers, we obtain the representation of the bipartite graph J, i.e.:

$$p_u^J = e_u^{(0)} + \cdots + e_u^{(K)}, p_i^J = e_i^{(0)} + \cdots + e_i^{(K)} \tag{15}$$

Similarly, we can obtain the training results $p_u^{G'}$ and $p_v^{G'}$ of GCN in the user-entity subgraph G' and the training results p_i^g and p_v^g of GCN in the project-entity subgraph g.

Finally, we construct comparison losses based on u, i, and v, respectively, as follows:

$$L^u = -\log \frac{e^{s(p_u^J, p_i^J)/\tau}}{e^{s(p_u^J, p_i^J)/\tau} + \sum_{k\neq i} e^{s(p_u^J, p_k^J)/\tau} + \sum_{k\neq i} e^{s(p_u^J, p_k^g)/\tau}} \tag{16}$$

where r denotes other project nodes in the user-project bipartite graph J and the project-entity subgraph g. That is:

$$L^i = -\log \frac{e^{s(p_i^J, p_u^{G'})/\tau}}{e^{s(p_i^J, p_u^{G'})/\tau} + \sum_{k \neq u} e^{s(p_i^J, p_k^J)/\tau} + \sum_{k \neq u} e^{s(p_i^J, p_k^{G'})/\tau}} \tag{17}$$

where r denotes other project nodes in the user-project bipartite graph J and the user-entity subgraph G'. That is:

$$L^v = -\log \frac{e^{s\left(p_v^g, p_v^{G'}\right)/\tau}}{e^{s\left(p_v^g, p_v^{G'}\right)/\tau} + \sum_{k \neq i} e^{s\left(p_v^g, p_k^g\right)/\tau} + \sum_{k \neq i} e^{s\left(p_v^g, p_k^{G'}\right)/\tau}} \tag{18}$$

Here, r denotes other project nodes in the user-entity subgraph G' and the project-entity subgraph g. $s(\ )$ denotes the calculation of cosine similarity, and τ is the temperature coefficient. The above three equations respectively represent the optimization of the representations of users, projects, and entities, with the aim of reducing the distance between the same users, projects, and entities in the Euclidean space under different subgraph views, while increasing the distance between other nodes under the same view and other views.

Finally, the inner product between the project and user representations is calculated as the prediction result, i.e.:

$$p_u = p_u^J \oplus p_u^{G'} \tag{19}$$

$$p_i = p_i^J \oplus p_i^g \tag{20}$$

$$\hat{y}_{(u,i)} = p_u^T p_i^g \tag{21}$$

Consistent with other recommended methods, use BPR loss to fit the interaction data, i.e.:

$$L_{BPR} = \sum_{(u,i,j) \in O} -\ln \sigma\left(\hat{y}_{(u,i)} - \hat{y}_{(u,j)}\right) \tag{22}$$

where (u,i) denotes an observed interaction and (u,j) denotes an unobserved interaction. The total loss is defined as the sum of the BPR loss and the contrast loss, i.e.:

$$L = L_{BPR} + L^u + L^v + L^i \tag{23}$$

The optimization objective is to minimize the loss function using the Adam optimizer.

IV. Recommendation effectiveness based on knowledge graphs

In recent years, with the widespread application of information technology and digital technology, online learning has become an important method of daily learning. Online learning has lowered the threshold for acquiring knowledge, improved the ability to utilize educational resources, and greatly promoted educational equity. However, with the rapid increase in the number of online courses and course resources, some issues that need to be addressed urgently have also emerged. This chapter primarily focuses on validating and analyzing the course knowledge recommendation model constructed in the preceding sections, with the aim of providing new methods for the effective integration of online course resources.

IV. A. Verification of the effectiveness of course knowledge recommendations

IV. A. 1) Model Performance Comparison

To validate the effectiveness of the course knowledge recommendation model designed in this paper, a comparative experiment was designed. The experimental comparison methods primarily include the AROLS method combining learning style models and collaborative filtering, the DNN method based on deep neural networks, the UserCF method based on user-based collaborative filtering, and the ItemCF method based on item-based collaborative filtering. This paper uses accuracy, recall, and F1 score as evaluation metrics, and selects the knowledge graph constructed in the previous section as the data source. The comparison experiment results for accuracy, recall, and F1 score are shown in Figures 4 to 6.

Based on the accuracy comparison results in Figure 4, it can be seen that the course knowledge recommendation model designed in this paper has a significant improvement in accuracy compared to the traditional ItemCF method, UserCF method, and DNN method based on deep neural networks. For example, when the TopN recommendation value is 10, the average accuracy rate of the method proposed in this paper is 0.308, while the lowest accuracy rate of the UserCF method is 0.097. However, compared with the KG-SLSTM-GCN model and the collaborative filtering recommendation method AROLS optimized by learning style, the accuracy rate is improved by an average of 20.31%. This is because the KG-SLSTM-GCN model utilizes a knowledge graph, which contains background information on course knowledge and the relationships between knowledge entities. It also integrates a learning knowledge network composed of student behavior data, expanding the hidden associations between students and courses and supplementing the interaction data between students and courses, thereby further enhancing recommendation effectiveness.

As shown in the recall rate comparison results in Figure 5, as the TopN value increases, the recall rates of all comparison methods gradually increase, and ItemCF and UserCF perform worse than the DNN method in terms of recall rate. Due to the unique graph network structure of knowledge graphs, they can identify weak associations within the graph and uncover potential association paths, resulting in a significant improvement in recall rate compared to existing methods. Therefore, the KG-SLSTM-GCN model achieves an average 70.15% improvement in recall rate compared to the DNN method.

In the F1 score comparison results, ItemCF and UserCF show no significant difference in overall performance. The DNN method outperforms the other three existing methods in terms of F1 score, which may be due to the DNN's ability to better uncover hidden features between elements in multi-feature nonlinear data and utilize multi-layer networks to fit these hidden mappings, demonstrating strong overall performance. A knowledge graph is a large graph formed by integrating various types of heterogeneous auxiliary information, while leveraging the advantages of graph network structures in association matching and path analysis. Therefore, the KG-SLSTM-GCN model shows a significant improvement in F1 score compared to other methods, with a 34.82% increase over the DNN method.

In summary, the course knowledge recommendation model constructed by combining knowledge graphs, stacked LSTM, and GCN demonstrates good performance, providing support for the construction of course knowledge ontologies and knowledge mapping, and further enhancing the effectiveness of online education.

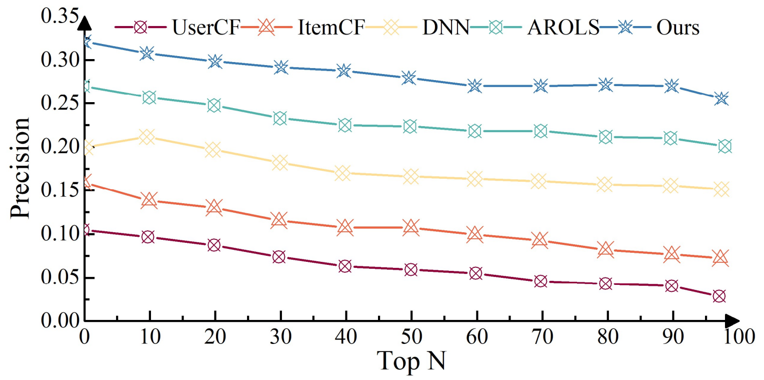


Figure 4: Precision comparison results

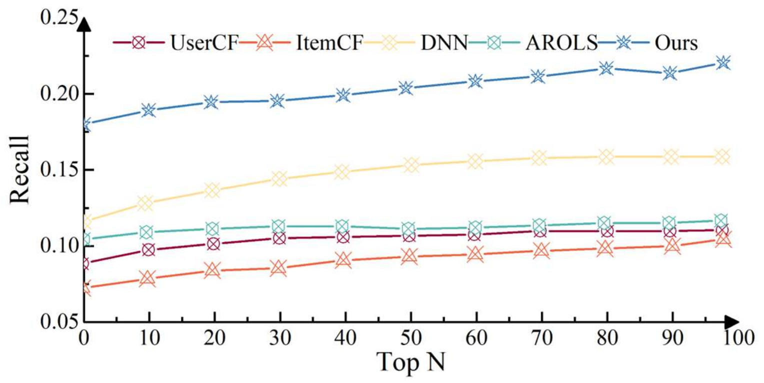


Figure 5: Recall comparison results

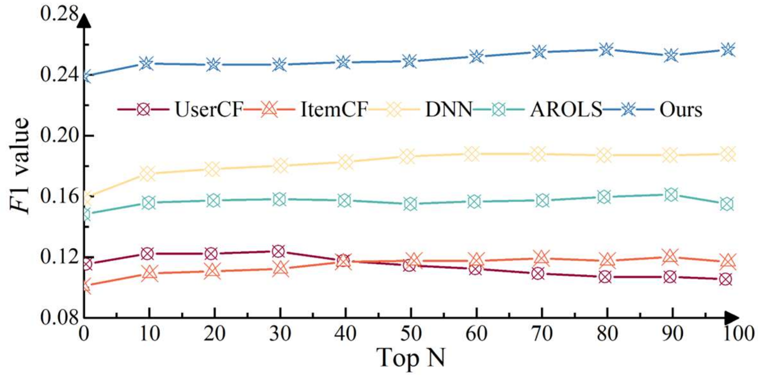


Figure 6: F1 value comparison results

IV. A. 2) Ablation experiments of the model

The course knowledge recommendation model KG-SLSTM-GCN established in this paper is mainly composed of three modules: knowledge graph, stacked LSTM, and GCN. To evaluate the impact of different modules in this model on course knowledge recommendation performance, this section designs ablation experiments for several modules in the model. By removing a module from the recommendation model, the necessity of each module is tested. Based on the course knowledge dataset constructed in the preceding section, this paper additionally selected an online education-related dataset for comparison. For evaluating the model's recommendation performance, this paper selected HR@K and MAP@K. Here, HR and MAP represent click-through rate and mean average precision, respectively, and K denotes the length of the recommendation list provided to the user. The results of the ablation experiments on different datasets are shown in Tables 1 and 2.

As shown by the experimental results, in the ablation experiments for the KG-SLSTM-GCN model, removing any single module significantly impacts the course knowledge recommendation performance of the KG-SLSTM-GCN model.

In the KG-SLSTM-GCN model without the knowledge graph, the auxiliary information interaction module is removed, and the user-item interaction module only uses the text embedding vectors of course titles. This model variant results in the most severe decline in recommendation performance, highlighting the importance of knowledge graph entity representations in improving the accuracy of course knowledge and user representations, and enhancing the effectiveness of course knowledge recommendations. In the KG-SLSTM-GCN model without introducing stacked LSTM to map knowledge graph entity embeddings to word embedding space, the recommendation performance also shows a certain decline. This indicates that the alignment of entity embedding space and word embedding space is also crucial for ensuring the accuracy of entity and word representations. If graph convolutional neural networks are not used in the KG-SLSTM-GCN model, it will also cause some damage to the model accuracy. This demonstrates that incorporating GCN into course knowledge representation training helps the course knowledge representation better perceive the latent relationships between knowledge graph entities, thereby improving course knowledge recommendation performance.

Table 1: Results of ablation experiment (dataset: Knowledge graph)

Model	HR@5	HR@10	HR@15	MAP@5	MAP@10	MAP@15
Ours	**0.357**	**0.602**	**0.748**	**0.363**	**0.457**	**0.472**
Ours-KG	0.236	0.465	0.634	0.216	0.299	0.323
Ours-SLSTM	0.317	0.579	0.726	0.334	0.418	0.438
Ours-GCN	0.328	0.582	0.719	0.328	0.415	0.446

Table 2: Results of ablation experiment (dataset: Online education)

Model	HR@5	HR@10	HR@15	MAP@5	MAP@10	MAP@15
Ours	**0.345**	**0.648**	**0.792**	**0.349**	**0.456**	**0.481**
Ours-KG	0.256	0.521	0.675	0.258	0.348	0.373
Ours-SLSTM	0.327	0.626	0.773	0.326	0.435	0.458
Ours-GCN	0.313	0.615	0.761	0.315	0.411	0.432

IV. B. Data sparsity and parameter sensitivity

IV. B. 1) Data sparsity experiment

Through in-depth research and practical application of the KG-SLSTM-GCN model proposed in this paper, we further explored its performance under different ratios. We selected the area under the curve (AUC), accuracy (ACC), hit rate (HR), and normalized discounted cumulative gain (NDCG) as evaluation metrics. The performance trends of different metrics under different ratios are shown in Figure 7.

The experimental results show that as the ratio gradually increases, the model's performance in key evaluation metrics such as AUC, ACC, HR, and NDCG all exhibit an upward trend. This trend not only demonstrates the model's adaptability under different ratios but also shows that when the ratio reaches 100%, the model achieves peak performance across all metrics, demonstrating exceptional capabilities. The proposed KG-SLSTM-GCN model demonstrates its powerful recommendation capabilities across various scenarios and conditions, providing users with more precise and personalized recommendation experiences. Specifically, as the ratio increases from 10% to 100%, the model achieves average improvements of 2.97%, 5.48%, 18.12%, and 59.86% in AUC, ACC, HR, and NDCG, respectively. This result not only demonstrates the effectiveness of the proposed model in course knowledge recommendation tasks but also highlights its superior performance in handling sparse data. Additionally, based on the trend of changes, it can be observed that the differences in AUC and ACC are relatively small, while the NDCG metric shows significant variation, indicating that data sparsity has a greater impact on this metric. However, overall, the proposed model performs well in handling sparse data. In practical applications, user-course interaction data is often highly sparse, and the model proposed in this paper can still perform well in such scenarios, which undoubtedly provides strong support for the practical application of recommendation systems.

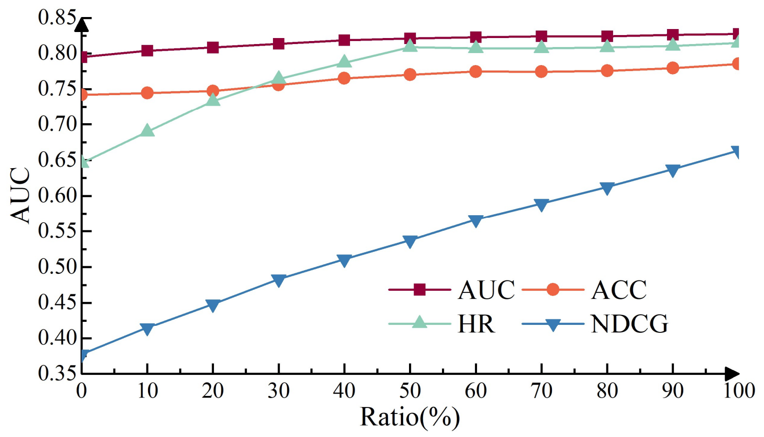
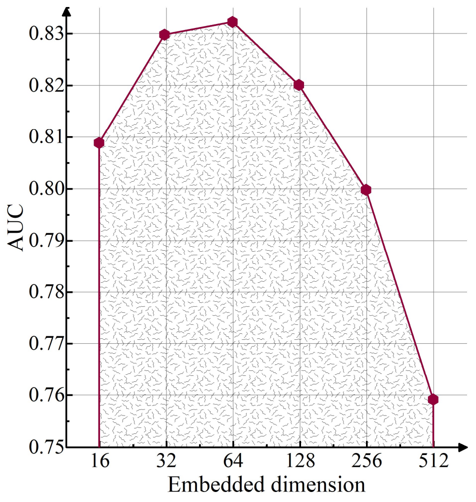


Figure 7: Trends of four indicators in different ratio

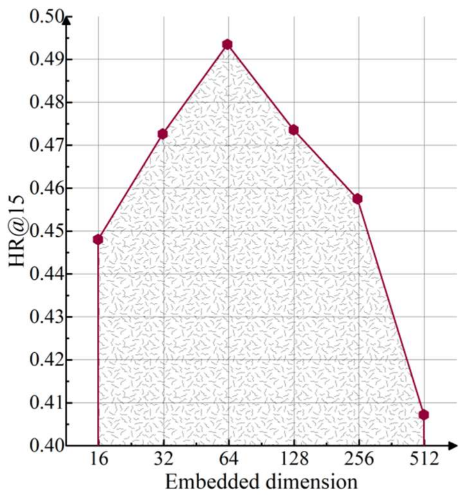
IV. B. 2) Parameter sensitivity experiment

For the KG-SLSTM-GCN model established in this paper, after satisfying the performance experiments in the previous section, this paper further analyzed the performance of AUC and HR@15 indicators under different hyperparameters. The hyperparameters of the KG-SLSTM-GCN model mainly include embedding dimension (D), neighbor sampling (K), and propagation layer number (L), which were verified on the course knowledge graph dataset constructed in the previous section.

Figure 8 shows the results of AUC and HR@15 for the KG-SLSTM-GCN model under different embedding dimensions. Observation reveals that increasing the dimension within a certain range can improve the performance of DKCR, but when the embedding dimension is too large, performance actually decreases. This is because while increasing the embedding dimension allows the KG-SLSTM-GCN model to encode more information, an excessively large embedding dimension may lead to overfitting of the KG-SLSTM-GCN model.



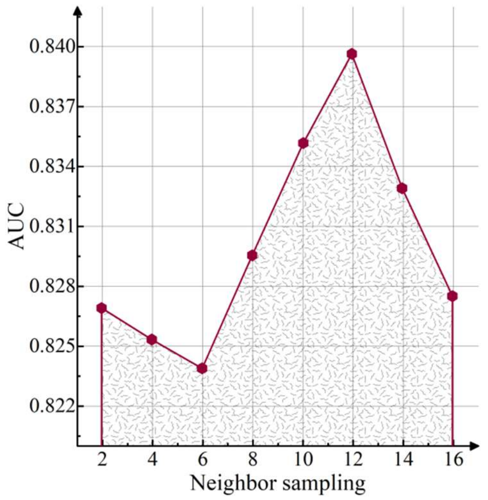
(a) AUC



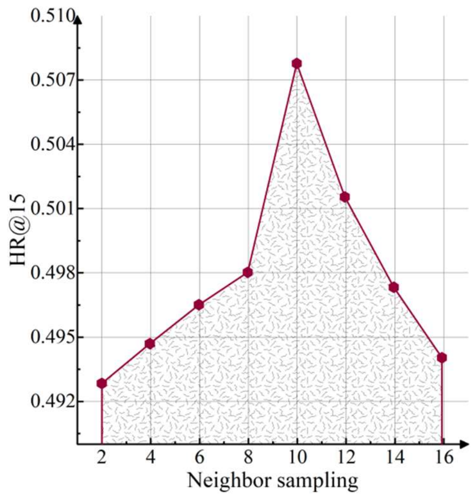
(b) HR@15

Figure 8: AUC and HR@15 under different embedding dimensions

Figure 9 shows the AUC and HR@15 results of the KG-SLSTM-GCN model under different neighborhood sampling sizes. It can be observed that increasing the neighborhood sampling size within a certain range can effectively improve the model's performance. However, an excessively large neighborhood sampling size not only introduces noise, leading to a decline in model performance, but also increases the model's training time. When the neighborhood sampling size is set to 10–12, the model achieves the best results. This is because, when the neighborhood sampling size is set to 10–12, the model can encode information to the greatest extent possible without introducing noise.



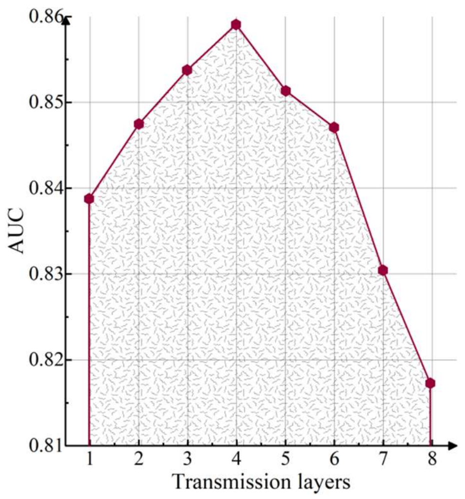
(a) AUC



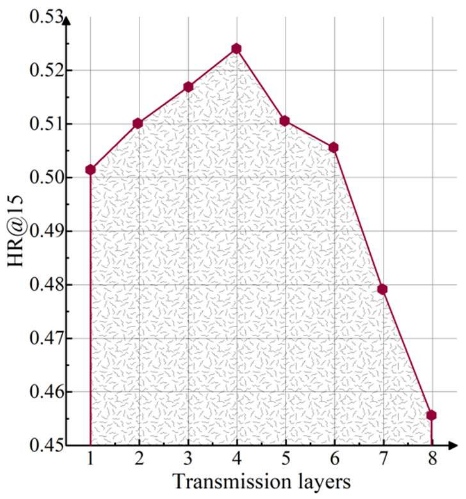
(b) HR@15

Figure 9: AUC and HR@15 under sampling from different neighbors

Figure 10 shows the AUC and HR@15 results of the KG-SLSTM-GCN model with different propagation layer numbers. It can be seen that as the number of propagation layers increases, the model can capture more information, but it also brings more noise. The model performance reaches its optimal level when the number of propagation layers is 4.



(a) AUC



(b) HR@15

Figure 10: AUC and HR@15 under different numbers of transmission layers

V. Conclusion

This paper takes course knowledge recommendation as its research object. Based on the establishment of a course knowledge ontology model, it combines stacked LSTM and GCN models to construct a course knowledge recommendation model and analyzes its performance through experiments. The experiments show that when the TopN recommendation value is 10, the average accuracy of the method proposed in this paper is 0.308, which is 1.18 times higher than that of the UserCF method. Furthermore, the knowledge graph, stacked LSTM, and GCN components in the model significantly influence the performance of course knowledge recommendation, effectively enhancing its effectiveness. By leveraging deep learning technology and knowledge graphs, data can be effectively mapped to construct a more precise knowledge ontology, thereby significantly improving the effectiveness of data recommendation.

References

[1] Kolovskaia, A. Y., & Ilin, A. D. (2023). Ontology of Artificial Intelligence as a Field of Engineering. Technology and Language, 4(4), 46.

[2] Hassan, B. A., & Rashid, T. A. (2021). Artificial intelligence algorithms for natural language processing and the semantic web ontology learning. arXiv preprint arXiv:2108.13772.

[3] Babaei Giglou, H., D'Souza, J., & Auer, S. (2023, October). LLMs4OL: Large language models for ontology learning. In International Semantic Web Conference (pp. 408-427). Cham: Springer Nature Switzerland.

[4] Schneider, P., Schopf, T., Vladika, J., Galkin, M., Simperl, E., & Matthes, F. (2022). A decade of knowledge graphs in natural language processing: A survey. arXiv preprint arXiv:2210.00105.

[5] Bhattarai, B., Granmo, O. C., & Jiao, L. (2023, March). An interpretable knowledge representation framework for natural language processing with cross-domain application. In European Conference on Information Retrieval (pp. 167-181). Cham: Springer Nature Switzerland.

[6] Ma, C., & Molnár, B. (2022). Ontology learning from relational database: Opportunities for semantic information integration. Vietnam Journal of Computer Science, 9(01), 31-57.

[7] Schilling, M., Bayerlein, B., von Hartrott, P., Waitelonis, J., Birkholz, H., Dolabella Portella, P., & Skrotzki, B. (2025). FAIR and Structured Data: A Domain Ontology Aligned with Standard-Compliant Tensile Testing. Advanced Engineering Materials, 27(8), 2400138.

[8] Arbaaeen, A., & Shah, A. (2021). Ontology-based approach to semantically enhanced question answering for closed domain: A review. Information, 12(5), 200.

[9] Mohammadi, S., Tavakolan, M., & Zahraie, B. (2018). An ontological framework for identification of construction resources using BIM information. In Construction Research Congress 2018 (pp. 694-704).

[10] Zouaq, A., Gagnon, M., & Ozell, B. (2010). Semantic analysis using dependency-based grammars and upper-level ontologies. Int. J. Comput. Linguistics Appl., 1(1-2), 85-101.

[11] Li, S., He, Z., & Wu, J. (2014, September). An ontology semantic tree based natural language interface. In The 9th International Symposium on Chinese Spoken Language Processing (pp. 226-230). IEEE.

[12] Hou, X., Ong, S. K., Nee, A. Y. C., Zhang, X. T., & Liu, W. J. (2011). GRAONTO: A graph-based approach for automatic construction of domain ontology. Expert systems with Applications, 38(9), 11958-11975.

[13] Al-Aswadi, F. N., Chan, H. Y., & Gan, K. H. (2020). Automatic ontology construction from text: a review from shallow to deep learning trend. Artificial Intelligence Review, 53(6), 3901-3928.

[14] Sicilia, Á., Nemirovski, G., & Nolle, A. (2017). Map-On: A web-based editor for visual ontology mapping. Semantic Web, 8(6), 969-980.

[15] Iglesias-Molina, A., Cimmino, A., Ruckhaus, E., Chaves-Fraga, D., García-Castro, R., & Corcho, O. (2024). An ontological approach for representing declarative mapping languages. Semantic Web, 15(1), 191-221.

[16] Gould, N., & Mackaness, W. (2016). From taxonomies to ontologies: Formalizing generalization knowledge for on-demand mapping. Cartography and Geographic Information Science, 43(3), 208-222.

[17] Zhao, M., Zhang, S., Li, W., & Chen, G. (2018). Matching biomedical ontologies based on formal concept analysis. Journal of biomedical semantics, 9, 1-27.

[18] Du, G. (2024). Research advanced in Chinese word segmentation methods and challenges. Applied and Computational Engineering, 37, 16-22.

[19] Huang, K., Xiao, K., Mo, F., Jin, B., Liu, Z., & Huang, D. (2021). Domain-aware word segmentation for Chinese language: A document-level context-aware model. Transactions on Asian and Low-Resource Language Information Processing, 21(2), 1-16.

[20] Sun, Q., Xu, J., Duan, Y., Zhang, P., Jiang, N., Alsoud, A. R., & Abualigah, L. (2024). Improving word similarity computation accuracy by multiple parameter optimization based on ontology knowledge. Multimedia Tools and Applications, 83(6), 17469-17489.

[21] Liu, J. N., He, Y. L., Lim, E. H., & Wang, X. Z. (2012). A new method for knowledge and information management domain ontology graph model. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 43(1), 115-127.

[22] Yin, P., Wang, H., & Guo, K. (2013). Feature–opinion pair identification of product reviews in Chinese: a domain ontology modeling method. New Review of Hypermedia and Multimedia, 19(1), 3-24.

[23] Li, Z., He, L., & Gao, D. (2022). Ontology construction and evaluation for chinese traditional culture: towards digital humanity. KO KNOWLEDGE ORGANIZATION, 49(1), 22-39.

[24] Liu, J., Zheng, B. J., Luo, L. M., Zhou, J. S., Zhang, Y., & Yu, Z. T. (2016). Ontology representation and mapping of common fuzzy knowledge. Neurocomputing, 215, 184-195.

[25] Xie, K., Wang, C., & Wang, P. (2021). A domain-independent ontology learning method based on transfer learning. Electronics, 10(16), 1911.

[26] Z. Chen,J. Chamorro Padial,J.M. López Gil,R.M. Gil & R. García. (2025). Tab2KGWiz: Interactive assistant for mapping tabular data to knowledge graphs. SoftwareX,31,102221-102221.

[27] Baoling Gui,Lydia Sam,Anshuman Bhardwaj,Diego Soto Gómez,Félix González Peñaloza,Manfred F. Buchroithner & David R. Green. (2025). SAGRNet: A novel object-based graph convolutional neural network for diverse vegetation cover classification in remotely-sensed imagery. ISPRS Journal of Photogrammetry and Remote Sensing,227,99-124.

[28] Jinxin Cao,Yiqiang Li,Yaqian Zhang,Xuechen Tang,Qihang Li,Yuling Zhang & Zheyu Liu. (2025). Long short-term memory neural network-decline curve analysis production forecast method for horizontal wells in tight reservoir based on sequence decomposition and reconstruction. Engineering Applications of Artificial Intelligence,158(PA),111482-111482.