GGNN+: A Graph Stacking Learning for Concept Prerequisite Prediction Based on Wikipedia Summary

Wei Dai[1], Kui Xiao[1*], Yamin Li[1*], Yan Zhang[1]

[1]School of Computer Science and Information Engineering, Hubei University, Wuhan, 430205, China.

*Corresponding author(s). E-mail(s): xiaokui@hubu.edu.cn; yamin.li@hubu.edu.cn;
Contributing authors: 1037530806@qq.com; zhangyan@hubu.edu.cn;

Abstract

The rapid development of internet technology has presented significant opportunities for online education. However, the abundance of intricate learning resources and the complex relationships among concepts on the internet also pose challenges for learners. Specifically, the prerequisite relationships among these concepts are of paramount importance in determining the sequence of learning for students. Consequently, the investigation of how to ascertain these prerequisite relationships between concepts has become one of the focal points of current research. With the emergence of graph neural networks, the field of artificial intelligence has made substantial progress. This development has also opened up new avenues and methods for the study of prerequisite relationships. In this context, our paper introduces an enhanced model utilizing a gated graph neural network approach for graph-based learning. Notably, we employ graph stacking for the first time within the gated graph neural network to update the representations of graph nodes. Subsequently, we construct a semantic heterogeneous graph for model learning. Following this, we employ an improved classification network for categorization. Ultimately, through the classification network, we can derive the prerequisite relationships between concepts. Indeed, experimental results on our dataset indicate that graph stacking learning, in comparison to the conventional gated graph sequence neural networks (GGNN) approach, demonstrates superior performance in terms of model efficacy.

Keywords: Graph stacking learning · GGNN+(our approach) · Prerequisite relationship · L-Siamese network · Mean aggregator

1 Introduction

The high-speed growth of technology and the onset of the information age have precipitated revolutionary changes in the field of education. Online education, distance learning, and the digital utilization of learning resources have seamlessly integrated into the contemporary education landscape.This educational paradigm shift not only enhances flexibility and convenience for both students and educators but also ushers in novel channels for knowledge dissemination and acquisition. However, given the diversity of learners' backgrounds, online students grapple with the challenge of prioritizing among a plethora of available resources.

Learning resources typically encompass various types, including online education platform videos, textbook knowledge points, Wikipedia articles, and course materials employed by instructors during lectures. Laurence and Margolis [1] in the field of cognition emphasize that the process of acquiring, applying, and generating knowledge inherently involves assessing conceptual prerequisites. Recognizing the growing significance of prerequisite relationships in education, scholars associated with Yu et al. [2] have explored a large-scale multi-relationship education dataset, MOOCCube, on the MOOC online course platform. This dataset offers a multi-faceted lens through which to examine online education. The advent of the information age and the surge in online learning participants, reaching 200 million globally by 2021, prompted the discovery of the MOOCCubeX dataset by Yu et al. [3]. This dataset not only encompasses diverse learning resources but also spans a broad spectrum of disciplines and topics. Nevertheless, learners frequently grapple with confusion when confronted by this abundance of educational resources.

Addressing this challenge typically necessitates the development of a well-defined course sequence, thereby enhancing the efficiency of online learning. Concepts and learning resources within a shared domain can be correlated within specific environments, So the prerequisite relationships that determine online learning resources are these concepts. In summary, concepts play a pivotal role in the prerequisite relationships among learning resources. Thus, in the pursuit of unveiling latent prerequisite relationships among concepts, Yang et al. [4] introduced a graph-based framework for depicting these interconnections, facilitating comprehension and organization of knowledge. Subsequently, Roy et al. [5] employed the PREREQ framework to deduce prerequisite relations among concepts.

This paper introduces an enhanced method for identifying prerequisite relations among concepts. Initially, a heterogeneous graph is constructed utilizing knowledge entries from Wikipedia. Subsequently, the enhanced model processes this heterogeneous graph, followed by prediction classification with an improved classifier, ultimately yielding the desired outcomes on the referenced dataset.

2 Related work

Exploring Precedence Relations among Concepts offers numerous valuable applications, such as personalized course guidance, optimized course design, and enhanced automated assessment systems for evaluating student knowledge. For instance, Xiao et al. [6] infers prerequisite relationships among concepts by extracting subtitles from

MOOC videos. Furthermore, we can uncover prerequisite relationships among concepts from learning resources, as demonstrated in [7]. This is achieved through the analysis of dependency data among university courses and the creation of concept maps that reveal prerequisite relations between these courses. Conversely, Manrique et al. [8] employs pr-page-rank (PageRank) [9] and Point-wise mutual information (PMI) [10] to construct conceptual maps for exploring precedence relations among learning resources. Additionally, Gasparett et al. [11] incorporates educational documents as a reference dataset for the study.

All of the above examples use machine learning and feature definitions to infer prerequisite relations between concepts. In addition, there is a group of researchers using graph neural networks, for example, Li et al. [12] proposes variational graph auto-encoders (VGAE) design for cross-domain prerequisite chain learning. Another study [13] combines relational graph convolutional network (RGCN) [14] and Bert [15] for embedding learning. Attention mechanisms, including multi-head attention, are frequently applied. A detailed discussion of multi-head attention mechanisms can be found in [16].

Nonetheless, there are typically multiple types of relationships between concepts, particularly within intricate heterogeneous graphs. Traditional graph neural networks like graph neural network (GNN) [17] and graph convolutional networks (GCN) [18] may struggle to handle these complex relationships effectively. Hence, this paper introduces an enhanced model known as GGNN+ based on the gated graph sequence neural networks (GGNN) [19] framework. We begin by constructing a semantic heterogeneous graph, denoted as G_s, comprising three distinct subgraphs: G_t (containing $TF-IDF$ information), G_r (comprising $RefD$ [20] data [Measuring prerequisite relations among concepts]), and G_l (composed of the longest sequence algorithm (LCS) [21]). Next, we feed the heterogeneous graphs and Bert pre-training vectors into GGNN+ for stacking learning on graph, subsequently fusing features through Mean Aggregator [22]. Finally, we employ the L-Siamese Network classifier to predict classifications, enabling the representation of potential relationships among concepts to be learned.

3 Problem definition

In this section, we will provide a detailed explanation of the symbol definitions used throughout the paper and present our proposed solution. To begin, we consider a set containing n concepts, denoted as $C = \{c_1, c_2, \ldots, c_n\}$, where each concept c_i represents an abstract idea or topic within the domain of knowledge. It is worth noting that we also introduce a special concept, namely S̈ummary ;̈ which serves to represent a summarization or abstraction of knowledge.

To describe the prerequisite relationship between concepts, we define a set of concept pairs $\mathbf{P} = \{\langle c_i, c_j\rangle | i \to j\}$, where $\langle c_i, c_j\rangle$ means that a concept c_i is a prerequisite for a concept c_j. This set $\mathbf{P}$ is employed to explicitly delineate the interdependencies among concepts, facilitating a deeper comprehension of the structure and dynamics of knowledge.

In mathematical terms, we can represent the prerequisite relationships among concepts as follows, Where $p(c_i, c_j) \in \mathbf{P}$:

$$p(c_i, c_j) = \begin{cases} 1, & c_i \text{ is a prerequisite of } c_j, \\ 0, & \text{else.} \end{cases} \tag{1}$$

Furthermore, we have constructed a heterogeneous semantic relation graph, denoted as $G_s(C_s, E_s)$, by incorporating the "Summary" concept. This graph serves to represent semantic relationships between concepts and plays a crucial role in capturing the semantic associations within knowledge, providing us with a framework for extracting prerequisite relationships among concepts.

4 The proposed method

4.1 The whole framework graph

Typically, in the field of education, we can organize concepts into a set, which in turn can correspond to a semantic graph within Wikipedia. To delve deeper into the semantic relationships between concepts, it becomes crucial to construct such a Wikipedia semantic graph. Currently, when constructing feature matrices for inter-concept relationships, many methods rely on manually defined generic features, such as article clickstreams [23] and article linking relationships [24], among others. However, this approach may overlook the underlying relationships between semantic matrices.

To address this issue, we propose a semantic relation heterogeneous graph based on Wikipedia articles, which we refer to as $G_s(C_s, E_s)$. This graph consists of three subgraphs: $G_t(C_t, E_t), G_r(C_r, E_r)$, and $G_l(C_l, E_l)$. We feed the G_s graph, along with feature vectors generated by the pre-trained Bert model, into the enhanced GGNN model (GGNN+). Subsequently, classification predictions are made using the improved L-Siamese network. The entire framework is illustrated in the Fig. 1:

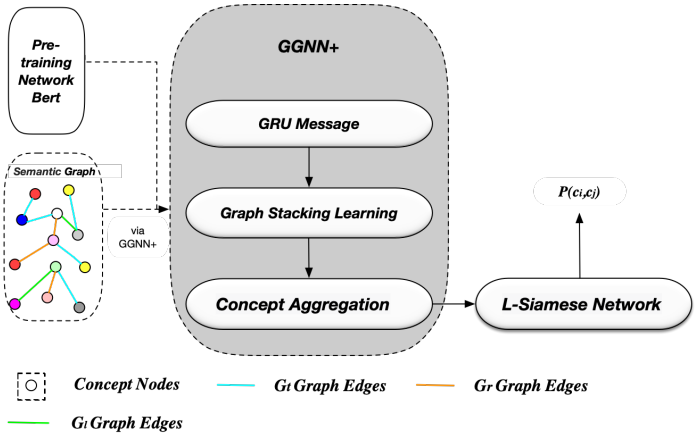

Fig. 1 The entire framework

4.2 Heterogeneous semantic graph construction

To delve deeper into the semantic relationship between two concepts, as depicted in Fig. 2, we utilize the abstract information from Wikipedia as our textual resource. In the field of text classification, these abstracts are commonly referred to as "Summary" and find extensive application across various academic disciplines. We regard the Wikipedia article summaries corresponding to each concept as our collection of textual resources, denoted as $Sum_s = \{Sum_1, Sum_2, Sum_3, ...Sum_n\}$, where n is the number of concepts. These concepts will be mapped into a vector space, represented as $C_s = \{C_1, C_2, C_3.....C_n\}$.We utilize these summaries as learning resources to construct three distinct types of semantic relationship graphs, named $G_t(C_t, E_t), G_r(C_r, E_r), \text{and} G_l(C_l, E_l)$. Furthermore, in this domain, numerous relevant studies have emphasized the significance of employing Wikipedia abstract information. For instance, Kaffee et al. [25] explores the use of Wikipedia summaries in text generation, while Gan et al. [26] investigates how leveraging Wikipedia-related data can enhance information retrieval tasks. These studies further validate the effectiveness of our approach and enrich the context of our research.

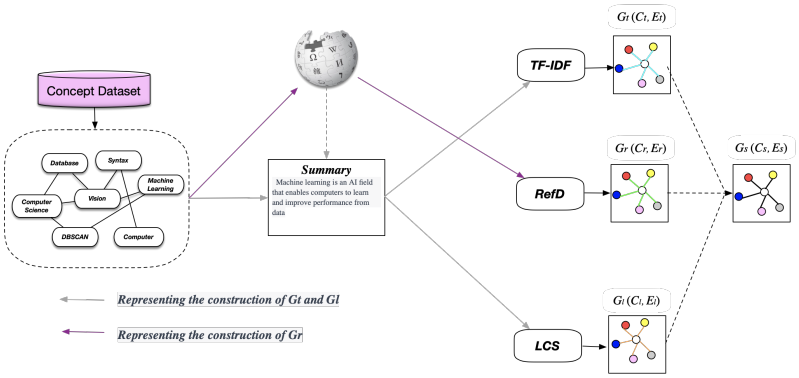

Fig. 2 Construction of the heterogeneous graph

Previous research has indicated that concepts may recurrently appear in different texts. To enhance the quality of node representation learning, we employ a cyclic addition operation applied to each subgraph extracted from the Summary. This addresses the issue of sparse matrices with fewer neighboring nodes.

$$A(i,j) = \begin{cases} TF-IDF(i,j) & \text{if } i \in Sum_s, j \in Sum_s \\ RefD(i,j) & \text{if } i \in C_s, j \in C_s \\ LCS(i,j) & \text{if } i \in Sum_s, j \in Sum_s \end{cases} \tag{2}$$

(1)**TF-IDF Weighted Edges Calculation**: We consider the Wikipedia concept's corresponding Summary entry as the core and utilize $TF-IDF$ to compute the weighted edges between them. Specifically, $TF-IDF(i,j) = Cos(TF-IDF(i), TF-IDF(j))$, where we take into account the indirect correlation between concept i in the same vector space and each word in $Sum(i)$. To achieve this, when calculating

$TF-IDF(i)$, we consider all words appearing in $Sum(i)$, thereby delving deeper into the relevant vocabulary within the Summary content. Finally, we apply the calculated $TF-IDF(i)$ and $TF-IDF(j)$ to the cosine function Cos to calculate the cosine value of the term frequency matrix, resulting in $TF-IDF(i,j)$.

(2)**Weighted Edges Based on Wikipedia Concept Relevance**: We observe that [20] employs a measurement method based on Wikipedia concept relevance. Given that Wikipedia concept entries are unique, we utilize $RefD$ to quantify the weight relationship between concepts i and j, selecting $TF-IDF$ from the two formulas of $w(c,A)$. Below are the two formulas of $w(c,A)$ and the formula for $RefD$:

$$\text{Ref } D(A,B) = \frac{\sum_{i=1}^{k} r\left(c_i,B\right)\cdot w\left(c_i,A\right)}{\sum_{i=1}^{k} w\left(c_i,A\right)} - \frac{\sum_{i=1}^{k} r\left(c_i,A\right)\cdot w\left(c_i,B\right)}{\sum_{i=1}^{k} w\left(c_i,B\right)} \tag{3}$$

EQUAL: A is represented by the concepts linked from it $(L(A))$ *with equal weights.*

$$w(c,A) = \begin{cases} 1 & \text{if } c \in L(A) \\ 0 & \text{if } c \notin L(A) \end{cases} \tag{4}$$

$TF-IDF$ *: A is represented by the concepts linked from it with* $TF-IDF$ *weights.*

$$w(c,A) = \begin{cases} tf(c,A) * \log \frac{N}{df(c)} & \text{if } c \in L(A) \\ 0 & \text{if } c \notin L(A) \end{cases} \tag{5}$$

where $C = \{c_1, \ldots, c_k\}$ is the concept space; $w(c_i, A)$ weights the importance of c_i to A; $r(c_i, A)$ is an indicator showing whether c_i refers to A, possibly a link in Wikipedia, a mention in a book, a citation in a paper, etc.

(3)**Longest Common Subsequence Algorithm (LCS)**: To compute the weighted edge between concept i and concept j, we employ the conventional LCS algorithm. The LCS algorithm is utilized for determining the longest common subsequence between two text sequences. Despite its relative simplicity, the LCS algorithm proves highly effective in assessing textual similarity. For instance, in the case of the strings "ABCD" and "ACDF," their longest common subsequence is "ACD" since both strings comprise ordered sequences of elements. Hence, we opt for the dynamic LCS planning algorithm, establishing a two-dimensional matrix where x_i represents the $i-th$ element of the first sequence, and y_j represents the $j-th$ element of the second sequence. Subsequently, by comparing the elements of the sequences and populating the table, we eventually obtain the longest common subsequence matrix, denoted as $LCS(i,j)$. The algorithm is outlined as follows:

$$LCS(i,j) = \begin{cases} 0 & \text{if } i=0, j=0 \\ LCS[i-1,j-1]+1 & \text{if } i,j>0 \text{ and } x_i = y_j \\ \max(LCS[i,j-1], LCS[i-1,j]) & \text{if } i,j>0 \text{ and } x_i \neq y_j \end{cases} \tag{6}$$

4.3 Concept representation via GGNN+

The GGNN model is an end-to-end graph learning model that updates node representations based on the dynamic graph structure at each time step and captures long-range dependencies between nodes through message passing. However, it's noteworthy that GGNN primarily emphasizes information aggregation from neighboring nodes, potentially overlooking graph-to-graph connections. Drawing inspiration from inductive learning methods like GraphSage [22], we opted for an approach involving graph-by-graph neighbor aggregation. In this method, the representation of one graph in each layer serves as the feature input for the subsequent graph. This approach not only preserves the functionality of the gated recurrent unit (GRU) [27] gating unit, but also utilises the representation of each graph aggregated by the GRU as input for the next graph.

Below, you can see the workflow diagram of the enhanced GGNN+ model, as depicted in Fig. 3:

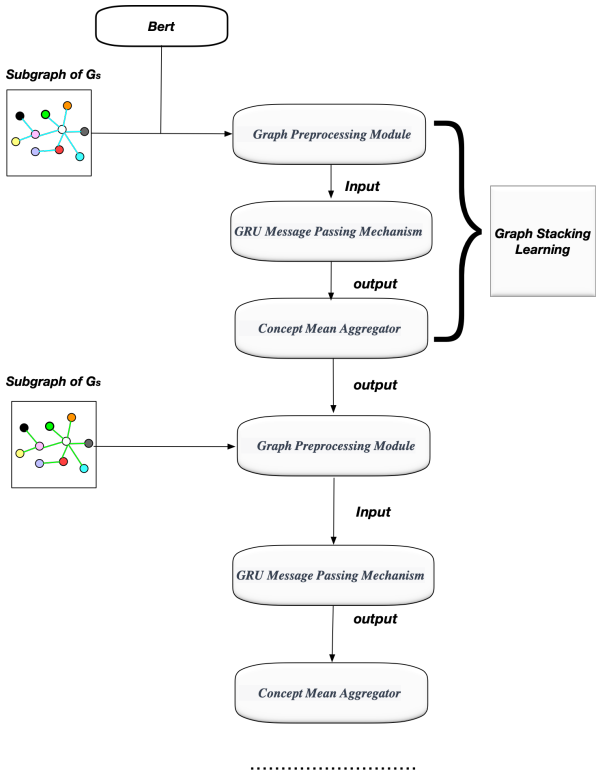

Fig. 3 The workflow of the GGNN+ model

Graph Preprocessing Before processing the graph, each concept is projected into a lower-dimensional space $c \in \nu$, $[h_1^{g-1}, h_2^{g-1}, h_3^{g-1}, \ldots, h_{[\nu]}^{g-1}]$ which is a vector stacked with $R^{d \times d}$ corresponding to represents a graph node representation of the

previous layer. And $[h_1^g, h_2^g, h_3^g, \ldots, h_{[\nu]}^g]$ is also a vector stacked with $R^{d\times d}$ to represent the pre-representation of the graphs that are ready to be output by graph stacking learning in this layer. For the update representation of each subgraph in G_s, we adhere to the update rules of GGNN, setting the connection matrix X as X_i^I and X_i^O, both serving as learning matrices for updates, as illustrated in Fig. 4.

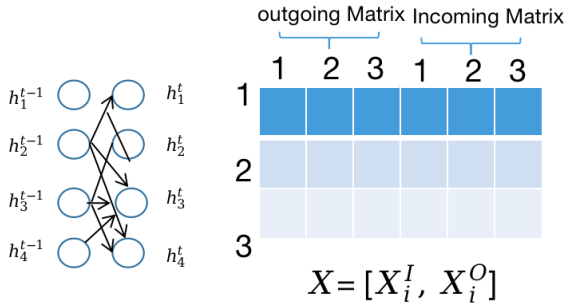

Fig. 4 Graph preprocessing of matrix

$$a_i^g = \text{Concat}\left(X_i^I \left[h_1^g, \ldots, h_{[\nu]}^g\right] W_x^I + b^I, X_i^O \left[h_1^{g-1}, \ldots, h_{[\nu]}^{g-1}\right] W_x^O + b^O\right) \quad (7)$$

The above formula illustrates a preprocessing step of concept graphs in GGNN. Due to the heterogeneity of the graph, we use $g \in [0, 1, 2]$ as indices for each update pass through the graph, rather than time steps. $h_i^0 \in R^{1\times d}$ represents the initial input value, acquired by multiplying the feature matrix i obtained from Bert with the adjacency matrix of the first layer of the map. To distinguish between the input and output layers, we designate $X_i^I \in R^{d\times d}$ as the connection matrix for the input layer, and $X_i^O \in R^{d\times d}$ as the connection matrix for the output layer. Additionally, $W_x^I \in R^{d\times d}$ and $W_x^O \in R^{d\times d}$ as weight matrices for the input and output layers, while $b^I \in R^d$ and b^O represent bias functions for the input and output layers. Finally, $a_i^g \in R^{2d\times 2d}$ is used to extract the neighbor node information of the subgraph.

Graph Stacking Learning To more effectively aggregate neighboring concept nodes, GGNN+ employs the update pattern of GRU. Subsequently, we introduce two gates, namely the update gate and reset gate, to determine whether the current node should consider information from adjacent nodes, as depicted in Fig. 5. We then utilize the update gate to compute the new concept representation. It is noteworthy that each input involves global message passing across the entire graph, enabling each node to acquire feature representations of neighbors from the entire layer. In Fig. 5, it can be observed that we incorporate G_t, the input adjacency matrix for the first layer, along with Bert into GGNN+. Finally, GGNN+ utilizes the Mean Aggregator to summarize the node representations for one layer. Furthermore, we introduce G_r as the adjacency matrix for the second layer, and put the output of the first layer as the feature matrix along with the adjacency matrix of the second layer into GGNN+ to get the feature representation of the second layer of graph stacking learning. This

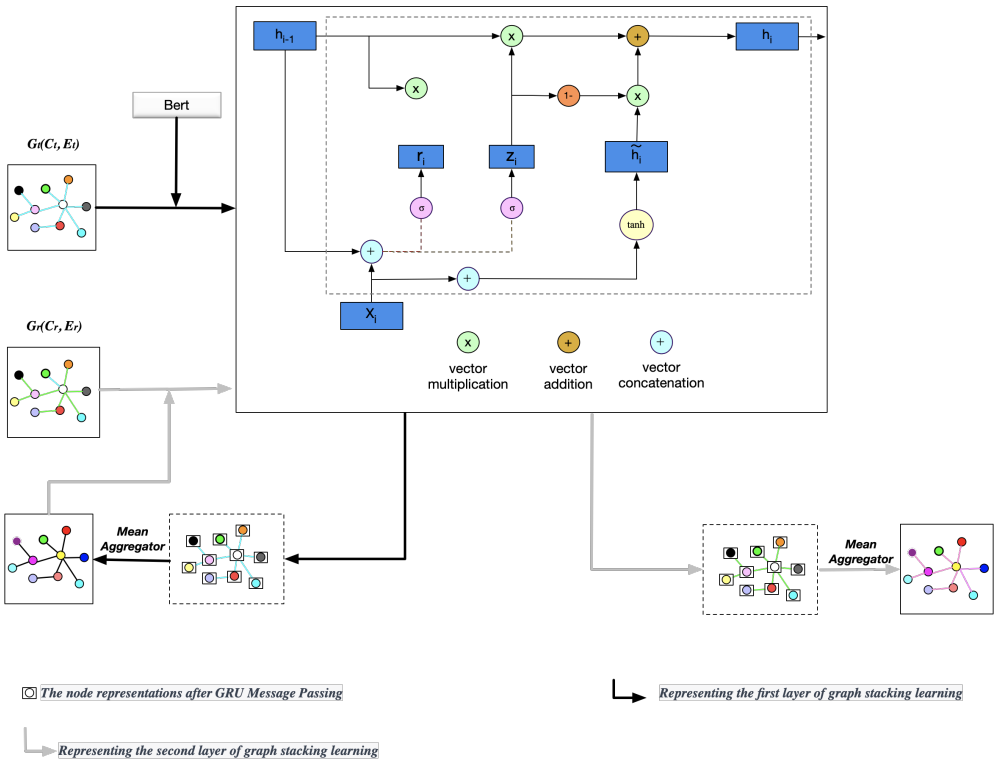

Fig. 5 The process of two-layer graph stacking learning

learning pattern is referred to as graph stacking learning within the GGNN+ model. Below is the mathematical representation of the process of updating concept nodes within a subgraph in the heterogeneous graph G_s.

$$\begin{aligned} z_i^g &= \sigma\left(W_y a_i^g + U_z h_i^g\right), \\ r_i^g &= \sigma\left(W_{\mathrm{r}} a_i^g + U_r h_i^g\right), \\ \tilde{h}_i^g &= \tanh\left(W_{\mathrm{k}} a_i^g + U_{\mathrm{k}}\left(r_i^g \odot h_i^{g-1}\right)\right) \\ h_i^g &= \left(1 - z_i^g\right) \odot h_i^{g-1} + z_i^g \odot \tilde{h}_{i'}^g \end{aligned} \tag{8}$$

where $a_i^g \in R^{d\times d}$ represents the feature representation initialized by subgraphs from the heterogeneous graph and fed into GGNN. To learn the neighbourhood feature representation of a node, we define the update gate as z_i^g and the reset gate as r_i^g. $W_y, W_r, W_k \in R^{d\times 2d}$ and $U_z, U_r, U_k \in R^{d\times d}$ are learnable parameters, σ represents the sigmoid function, and $\odot$ denotes element-wise multiplication in vectors. tanh corresponds to the hyperbolic tangent function, while h_i^g serves as the output representation for graph stacking learning in the t-th layer.

Concept Feature Aggregation Following graph stacking learning, concept representations may still harbor latent relational information. In this context, the Mean

Aggregator, a commonly employed technique in graph neural networks, is typically used to aggregate the features of neighboring nodes. The application of this method in this context aims to efficiently consolidate information among neighboring nodes, providing us with a way to uncover potential prerequisite relationships between concepts.

This process involves integrating contextual information from nodes to gain a more comprehensive understanding of each concept. The utilization of Mean Aggregator aids in extracting information regarding concept relationships, including potential prerequisite relationships, from these integrated features. Therefore, Mean Aggregator serves as a crucial information aggregation tool for mining and analyzing associative relationships among concepts in graph data. The formula for Mean Aggregator is as follows:

$$e_i = \sigma\left(\mathrm{W_s} \cdot \mathrm{Mean}\left\{h_i^{\mathrm{g}}\right\} \cup \left\{h_n^{\mathrm{g}}\right\}, \mathrm{n} \in \mathrm{N}\right) \tag{9}$$

Following graph stacking learning, vectors are represented as $h_i^g \in R^{d \times d}$, where h_n^g represents the feature representation of neighboring nodes, and N denotes the number of concepts. $h_i^g \in R^{d \times d}$ and h_n^g will be computed using the Mean function to obtain the average node representation for each concept, resulting in each concept being fully integrated with a comprehensive feature representation. Within the Mean function, we begin by stacking the vectors learned from three different graphs, followed by row-wise averaging of these vectors. In other words, we aggregate the feature vectors of neighboring nodes by averaging each row of the final vector. During this process, we utilize the connection matrix $W_s \in R^{d \times d}$ as the aggregation weights for the Mean function. After the final processing by the Mean function, our resulting concept vector ensemble has adequately captured the respective feature information. Consequently, the set of feature vectors e_i is what we end up getting by learning all the subgraphs through graph stacking. This process helps to synthesise information from different graphs, allowing us to get a more comprehensive picture of the feature representation of each concept. This is especially important for dealing with complex heterogeneous graph data.

4.4 The classification of conceptual prerequisite relationships

Classification predictions were made using the L-Siamese Network, which incorporates $L1$ regularization into the input layer of the Siamese Network [28], as depicted in Fig. 6. According to [29], adjusting the hyperparameters of $L1$ regularization allows for the selection of the most influential features for model performance. In summary, $L1$ regularization induces sparsity and can drive model coefficients to zero, facilitating feature selection. In our experiments, we observed that the Siamese Network with the $L1$ paradigm exhibits greater stability and improved generalization performance compared to the Siamese Network without the $L1$ paradigm.

Initially, the concept vectors learned from GGNN+ are fed into a fully connected layer followed by a ReLU activation function $\tilde{e_i} = \mathrm{ReLU}\left(W \cdot e_i + b\right)$. Subsequently, we proceed to an $L1$ regularization layer, yielding the output $\overrightarrow{e_i} = \sum_{i=1}^{n} e_i$. Finally, we enter the Siamese Network for vector concatenation and accumulation, formally represented as $\dot{e} = [\overrightarrow{e_i}; \overrightarrow{e_i}; \overrightarrow{e_i}; \overrightarrow{e_i}; \overrightarrow{e_i} \otimes \overrightarrow{e_i}]$. And then, W_h^T represents

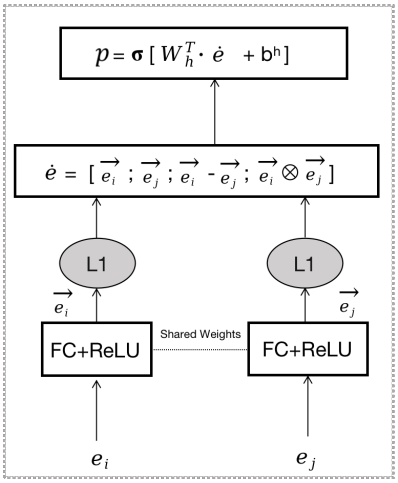

Fig. 6 L-Siamese network

the learning weights of the L-Siamese Network, b^h is the bias function, and p is the probability obtained through the σ activation function denoted as $p(c_i, c_j) = \sigma\left[W_h^T \cdot \dot{e} + b^h\right]$. Ultimately, we employ cross-entropy as the loss function $L_h = \frac{1}{|T|} \sum_{(c_i,c_j,y_{ij})\in T} - [y_{ij} \log (p\,(c_i, c_j) + (1 - y_{ij}) \cdot \log (1 - p\,(c_i, c_j)]$, with T denoting the training dataset, and $y_{ij} \in \{0, 1\}$ indicating whether c_i is a prerequisite for c_j.

5 Experiment result and discussion

5.1 Experiment setting

To validate the performance of our model, the datasets we used are listed in Table 1:

Table 1 Dataset performance

Dataset	#Concepts	#Pairs
Data_mining	120	826
Geometry	89	1681
Physics	153	1962
Precalculus	223	2060
University Course	365	1008
W-ML	120	486

AL-CPL: AL-CPL is a dataset derived from the paper [30]. This dataset spans four distinct domains: Data_mining, Geometry, Physics, and Precalculus. The Data_mining dataset comprises 826 concept pairs, with 292 of them labeled as positive samples. In Geometry, there are 1681 concept pairs, of which 524 are positive samples. The Physics dataset contains 1962 concept pairs, with 487 being positive samples. Lastly, the Precalculus dataset includes 2060 concept pairs, with 699 marked as positive

samples. It's worth noting that these datasets were collected from textbooks across these four domains and meticulously labeled by experts. The primary research goal of these datasets is to explore active learning methods in strict partial order relationships.

University Course: This dataset [5] includes 654 courses from various American universities, comprising 861 prerequisite relationships among 1008 concept pairs, manually annotated.

W-ML: For the W-ML dataset, we utilized MOOC data mentioned in [2]. This dataset covers two domains: Wikipedia data structures and algorithms (W-DSA) and Wikipedia machine learning (W-ML). In the W-ML domain, the dataset consists of 548 course videos. The W-ML dataset is derived from 240 courses and includes 120 concepts along with corresponding videos, gathered from Wikipedia and the 240 courses.

Across all datasets, only concepts are labeled. We split the dataset into a 1/3 test set and a 2/3 training set. To address the issue of imbalanced positive and negative samples in the dataset, we performed edge flipping and oversampling. Specifically, we oversampled the positive samples in both the test and training sets by a factor of 3.5 and 1.5, respectively.

Regarding parameter settings, model parameters were initialized randomly from a Gaussian distribution with a mean of zero and a standard deviation of $\sigma = 0.3$. The initial learning rate for the Physics dataset was 0.06, while for other datasets, it was set at 0.01. The number of iterations for experimental results was set to 300, with a learning rate decay of 0.99 every 30 iterations. We used the stochastic gradient descent algorithm for training the model. Experimental results represent the average evaluation scores from five experiments when the loss value did not decrease for 20 epochs. All parameters in this experiment were empirically set.

For baseline models, we adjusted Support Vector Machine (SVM) [31], RefD, and other models accordingly. The parameter settings for heterogeneous graph attention network (HAN) [32], GGNN [33], and RGCN [34] were the same as those for our GGNN+ model. All experimental results have been validated. Additionally, we set the pre-trained vector size for the Bert model to 300 dimensions. After GGNN+ processing, the output dimension matches the number of concepts, while the L-Siamese Network's hidden layer is 64-dimensional. We configured the graph stacking learning as a one-time process, with each graph embedding size set to the number of concepts. It's worth noting that minor experimental manipulations did not significantly affect the experimental results. Detailed discussions on adjustments to certain models and graph stacking learning will be presented in subsequent chapters.

5.2 Baselines

In our experimental methodology, we investigated several existing baselines to assess their performance on the dataset. These baselines comprise:

SVM: SVM is a machine learning model. We utilized SVM for classification predictions by feeding it with models of our concept pairs.

RefD: RefD (Measuring prerequisite relations among concepts) is a link-based metric designed to measure pre-existing relationships among concepts. It offers a straightforward method to quantify the associations between concepts.

GAE: Graph Autoencoder (GAE) [35] is an unsupervised model that employs graph self-encoding. We input concept graphs into GAE for classification prediction.

VGAE: VGAE is an extension of GAE and was similarly employed in [36]. for learning concept prerequisite relations.

HAN: HAN in contrast to traditional GNN models, introduces an attention mechanism, enabling the model to focus on critical information, thereby improving performance. It has demonstrated exceptional results in text classification and sentiment analysis.

RGCN: RGCN is a model for handling multi-relational graph data, allows for learning diverse relationship types to better capture the interactions between entities.

GGNN: GGNN introduces gating mechanisms, akin to the gate units in GRU or LSTM, on top of traditional GNN models. These gate units enable the model to selectively update node information, thereby enhancing its ability to capture both local and global information in the graph.

By conducting experiments on our dataset with existing baselines, we obtained corresponding experimental outcomes. These results serve as a foundation for assessing the performance and effectiveness of our proposed approach. We utilized default parameters for the GGNN model.

5.3 Experiment results

In order to evaluate the performance of our model, we categorise our evaluation metrics into Precision (P), Recall (R), and F1-score (F) metrics, and from Table 2 we can see that our proposed GGNN+ approach is consistently outperforming the baseline approach.

Table 2 Evaluation Results

Dataset	Metric	SVM	RefD	GAE	VGAE	HAN	RGCN	GGNN	GGNN+
Data_mining	P	0.530	0.334	0.413	0.442	0.874	0.835	0.895	**0.904**
	R	0.700	0.469	0.629	0.102	0.870	0.249	0.925	**0.935**
	F	0.599	0.390	0.498	0.165	0.872	0.384	0.909	**0.919**
Geometry	P	0.694	0.470	0.253	0.517	0.913	0.914	0.922	**0.925**
	R	0.755	0.778	0.399	0.471	0.813	0.812	0.895	**0.912**
	F	0.723	0.586	0.310	0.487	0.860	0.860	0.908	**0.918**
Physics	P	0.541	0.378	0.265	0.638	0.828	0.799	0.865	**0.937**
	R	0.613	0.783	0.399	0.417	0.640	0.583	0.791	**0.942**
	F	0.575	0.510	0.319	0.541	0.722	0.674	0.826	**0.939**
Precalculus	P	0.667	0.578	0.362	0.329	0.913	0.883	0.914	**0.930**
	R	0.812	0.853	0.394	0.132	0.881	0.594	0.905	**0.934**
	F	0.733	0.689	0.378	0.188	0.897	0.710	0.910	**0.932**
University Course	P	0.552	0.631	0.482	0.533	0.844	0.767	0.959	**0.977**
	R	0.540	0.667	0.091	0.493	0.862	0.771	0.945	**0.962**
	F	0.538	0.649	0.153	0.512	0.852	0.768	0.952	**0.969**
W-ML	P	0.637	0.357	0.300	0.563	0.733	0.834	0.842	**0.867**
	R	0.638	0.475	0.691	0.716	0.730	0.776	0.842	**0.852**
	F	0.638	0.408	0.401	0.630	0.721	0.803	0.842	**0.859**

Due to slight differences in the inputs and outputs of baseline methods and our approach, we performed input normalization for the baseline methods. We partitioned the input graph edges of traditional machine learning methods SVM and RefD into five subsets for cross-validation, and the average of the results from these five runs served as the final outcome. Additionally, to evaluate the distinction between our approach and the unimproved GGNN method, we maintained consistent parameter settings for GGNN and GGNN+. As depicted in Table 2, the performance of our heterogeneous graph Gs improved consistently across various datasets after applying the GGNN+ method. For instance, in the Physics domain dataset of AL-CPL, GGNN+ yielded a 13% increase in the F1 score compared to the non-enhanced GGNN method. On other datasets such as Geometry, Data_ mining, and University Courses, GGNN+ also exhibited slight performance improvements. These outcomes suggest that the application of graph stacking learning in GGNN+ leads to enhanced effectiveness. To provide a more visual representation of experimental reliability, we present graphs illustrating the relationship between various metric scores and the number of Epoch iterations, as Fig. 7:

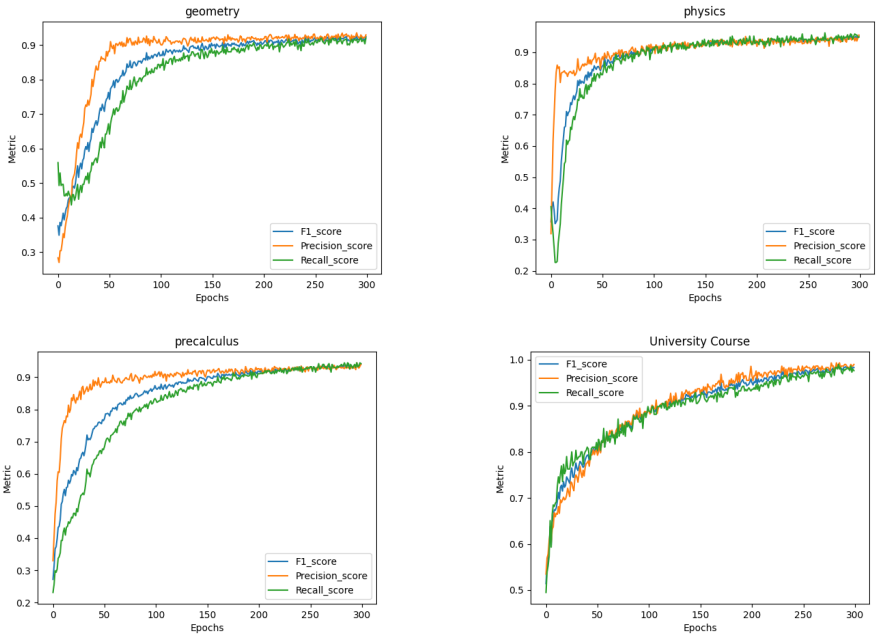

Fig. 7 Showing the performance metrics of the four domain datasets as the epoch changes

The line chart data in Fig. 7 above indicates that these four datasets have all reached a stable state after approximately 300 iterations, with the best performance achieved on the Precalculus dataset following 300 iterations. Furthermore, GGNN+ has demonstrated favorable results on the other datasets as well. Precision, Recall, and F1 scores across all four datasets show a gradual increase as the number of Epochs

rises, indicating the stability of the GGNN+ model and its compatibility with our existing datasets.

5.4 Experiment discussion

This section will discuss the compatibility of conceptual semantic graphs in the experiment and the feasibility study of the graph stacking learning method, in the experiment originally all parameters remain unchanged, we will increase the number of graph stacking learning as a way to test whether the number of graph stacking learning can affect the results of the experiment, the results of the test are as shown in the Table 3 under:

Table 3 The effect of the number of graph stacking learning on the experimental results

Dataset	Metric	1-layer	2-layer	3-layer
Data_mining	P	**0.904**	0.880	0.520
	R	**0.935**	0.918	0.512
	F	**0.919**	0.898	0.510
Geometry	P	**0.925**	0.917	0.273
	R	**0.912**	0.897	0.137
	F	**0.918**	0.907	0.159
Physics	P	**0.937**	0.933	0.894
	R	**0.942**	0.938	0.885
	F	**0.939**	0.936	0.889
Precalculus	P	**0.930**	0.911	0.862
	R	**0.934**	0.908	0.794
	F	**0.932**	0.910	0.832
University Course	P	**0.977**	0.921	0.506
	R	**0.962**	0.912	0.552
	F	**0.969**	0.916	0.534
W-ML	P	**0.867**	0.311	0.300
	R	**0.852**	0.171	0.691
	F	**0.859**	0.217	0.401

Table 3 shows that single-layer graph stacking learning yields the best performance in concept graphs, notably achieving an F1 score of 0.969 on the University Course dataset. This underscores the effectiveness of graph stacking learning in aggregating neighbor node information and enhancing model performance. However, it's crucial to exercise caution in selecting the number of layers and structure when designing graph stacking, as excessive stacking of identical graphs can have a counterproductive effect. Additionally, a series of experiments, referred to as semantic graph tests, were conducted to evaluate the impact of each layer of stacking on the experiments. In these experiments, each of the three semantic graphs was input separately, stacked once in the GGNN+ model for learning, and the corresponding F1 scores were obtained, as presented in the Table 4 below:

Based on the Table 4, we observe that the G_r graph performs well on three datasets: Data_mining, Geometry, and University Course. It's worth noting that the Gr graph is constructed based on relationships within Wikipedia articles. For instance, in the Data_mining dataset, concepts like ⟨DBSCAN, Arithmetic_mean⟩ suggesting that the

Table 4 Dataset performance

Dataset	Metric	G_t graph	G_r graph	G_l graph
Data_mining	F	0.916	**0.918**	0.917
Geometry	F	0.916	**0.919**	0.915
Physics	F	**0.944**	0.941	0.939
Precalculus	F	0.929	0.931	**0.932**
University Course	F	0.968	**0.969**	0.967
W-ML	F	0.837	0.853	**0.860**

G_r graph is more suitable for concept datasets within Wikipedia with abundant link relationships. On the other hand, the G_t graph exhibits superior performance on the Physics dataset, while the G_l graph achieves the highest F1 score in the Precalculus and W-ML datasets. This underscores our ability to optimize prediction results by tailoring concept representation training with different graphs for distinct datasets.

6 Conclusion and future work

In summary, this paper introduces GGNN+, a method aimed at addressing the issue of concept prerequisite relations in the field of education through graph stacking learning and L-Siamese Network. GGNN+ leverages the heterogeneous semantic graph Gs, constructed from Wikipedia and fundamental algorithms, to make classification predictions by learning relationships between concepts within this heterogeneous graph. It's noteworthy that the L-Siamese Network is an enhanced network that incorporates L1 regularization to improve model robustness and stability. Experimental results clearly demonstrate the significant impact of GGNN+ in the educational domain.

In summary, our contributions can be outlined as follows:

(i) A semantic heterogeneous graph is introduced to achieve multi-level understanding of the relationships among concepts.

(ii) The GGNN+ model for graph stacking learning is developed to more accurately learn the feature representation of the nodes.

(iii) We utilize the L-Siamese Network as a classification predictor to improve the robustness of the classification results.

In future research, scholars may consider the following directions:

(i)Constructing larger relational network datasets to further enhance model performance and accuracy.

(ii)Continuously refining graph stacking learning methods and conducting more experiments on large-scale datasets to gain deeper insights into their performance.

(iii)Exploring the use of open-source large-scale language models, such as LLM, for pretraining to enhance model representation and generalization capabilities.

It's worth noting that graph stacking learning methods have demonstrated exceptional performance in practical applications, particularly in handling large-scale dynamic graphs enriched with features and data, including time series, global information, and external data. Future studies can expand the model further to comprehensively explore concept prerequisite relations, providing more innovation and insights for the field of education.

Acknowledgments This work is supported by the National Natural Science Foundation of China (No.62377009, 62102136), Key R & D projects in Hubei Province (No.2021BAA188, 2021BAA184, 2022BAA044).

Authors contribution statement Wei Dai contributed to Conceptualization, Methodology, Software, Investigation, Formal Analysis, and Writing of the Original Draft;
Yamin Li and Kui Xiao contributed to this work in a manner similar to Wei Dai, involving Conceptualization, Methodology, Software, Investigation, Formal Analysis, and the Writing of the Original Draft;
Yan Zhang contributed to Funding Acquisition, Resources and Supervision.

Data availability and access We evaluate our method on the public AL-CPL, University Course, and W-ML datasets, which are each available at the following locations:

- AL-CPL dataset: https://github.com/harrylclc/AL-CPL-dataset
- University Course dataset: https://github.com/suderoy/PREREQ-IAAI-19
- W-ML dataset: http://keg.cs.tsinghua.edu.cn/jietang/software/acl17-prerequisite-relation.rar

If you need access to the code for the models in this article, please contact the corresponding author.

Declarations

Competing Interests The authors declare that they have no conflict of interest.

Ethical and informed consent for data used This paper does not include any studies involving human participants or animals conducted by the authors.

References

1. Laurence S, Margolis E (1999) Concepts and cognitive science. e margolis and s laurence concepts core readings https://doi.org/10.1002/wcs.137

2. Yu J, Luo G, Xiao T, et al (2020) Mooccube: A large-scale data repository for nlp applications in moocs. In: Meeting of the Association for Computational Linguistics

3. Yu J, Wang Y, Zhong Q, et al (2021) Mooccubex: A large knowledge-centered repository for adaptive learning in moocs. Association for Computing Machinery https://doi.org/10.1145/1122445.1122456

4. Yang Y, Liu H, Carbonell J, et al (2015) Concept graph learning from educational data. ACM https://doi.org/10.1145/2684822.2685292

5. Roy S, Madhyastha M, Lawrence S, et al (2018) Inferring concept prerequisite relations from online educational resources. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 9589-9594 https://doi.org/10.1609/aaai.v33i01.33019589

6. Xiao K, Bai Y, Wang S (2021) Mining precedence relations among lecture videos in moocs via concept prerequisite learning. Hindawi Limited https://doi.org/10.1155/2021/7655462

7. Chen L, Ye J, Wu Z, et al (2017) Recovering concept prerequisite relations from university course dependencies. In: Association for the Advancement of Artificial Intelligence

8. Manrique R, Sosa J, Marino O, et al (2018) Investigating learning resources precedence relations via concept prerequisite learning. In: 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI), https://doi.org/10.1109/WI.2018.00-89

9. Page L (1998) The pagerank citation ranking : Bringing order to the web. http://ilpubsstanfordedu:8090/422/1/1999-66pdf https://doi.org/10.1007/978-3-319-08789-4_10

10. Nelson DL, McEvoy CL, Schreiber TA (2004) Word associations: Norms, similarity, and frequency of occurrence. Memory & Cognition 32(7):1114–1129. https://doi.org/10.3758/BF03196959

11. Gasparetti F (2021) Discovering prerequisite relations from educational documents through word embeddings. Future Generation Computer Systems https://doi.org/10.1016/j.future.2021.08.021

12. Li I, Yan V, Li T, et al (2021) Unsupervised cross-domain prerequisite chain learning using variational graph autoencoders. In: Natural Language Processing, ACL, https://doi.org/10.18653/v1/2021.acl-short.127

13. Xu Y, Yang J (2019) Look again at the syntax: Relational graph convolutional network for gendered ambiguous pronoun resolution. In: Natural Language Processing, ACL, https://doi.org/10.18653/v1/W19-3814

14. Schlichtkrull M, Kipf TN, Bloem P, et al (2017) Modeling relational data with graph convolutional networks. https://doi.org/10.1007/978-3-319-93417-4_38, 1703.06103

15. Devlin J, Chang MW, Lee K, et al (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805 https://doi.org/10.48550/arXiv.1810.04805

16. Zhang J, Lan H, Yang X, et al (2022) Weakly supervised setting for learning concept prerequisite relations using multi-head attention variational graph auto-encoders. Knowledge-based systems p 247

17. Wu Z, Pan S, Chen F, et al (2021) A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems p 32

18. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. In: ICLR, https://doi.org/10.48550/arXiv.1609.02907

19. Li Y, Tarlow D, Brockschmidt M, et al (2016) Gated graph sequence neural networks. In: Proceedings of the International Conference on Learning Representations (ICLR)

20. Chen L, Wu Z, Huang W, et al (2015) Measuring prerequisite relations among concepts. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing

21. Hirschberg DS (1975) A linear space algorithm for computing maximal common subsequences. Communications of the ACM 18(6):341–343. https://doi.org/10.1145/360825.360861

22. Hamilton WL, Ying R, Leskovec J (2017) Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems

23. Hu C, Xiao K, Wang Z, et al (2021) Extracting prerequisite relations among wikipedia concepts using the clickstream data. In: Qiu H, Zhang C, Fei Z, et al (eds) Knowledge Science, Engineering and Management. Springer International Publishing, Cham, pp 13–26

24. Wen H, Zhu X, Zhang M, et al (2021) Combining wikipedia to identify prerequisite relations of concepts in moocs. In: Mantoro T, Lee M, Ayu MA, et al (eds) Neural Information Processing. Springer International Publishing, Cham, pp 739–747

25. Kaffee LA (2018) Learning to generate wikipedia summaries for underserved languages from wikidata. In: NAACL, https://doi.org/10.48550/arXiv.1803.07116

26. Gan L, Hong H (2015) Improving query expansion for information retrieval using wikipedia. international journal of database theory & application https://doi.org/10.14257/ijdta.2015.8.3.03

27. Cho K, van Merriënboer B, Gulcehre C, et al (2014) On the properties of neural machine translation: Encoder-decoder approaches. In: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation

28. ROOPAK, SHAH, EDUARD, et al (1993) Signature verification using a "siamese" time delay neural network. International Journal of Pattern Recognition and Artificial Intelligence 07(4):669–669

29. Tibshirani R (2011) Regression shrinkage and selection via the lasso: a retrospective. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73(3):267–288. https://doi.org/10.1111/j.1467-9868.2011.00771.x

30. Liang C, Ye J, Zhao H, et al (2018) Active learning of strict partial orders: A case study on concept prerequisite relations. In: Educational Data Mining, https://doi.org/10.48550/arXiv.1801.06481

31. Cortes C, Vapnik V (2009) Support-vector networks. Chemical Biology & Drug Design 297(3):273–297. https://doi.org/10.1007//BF00994018

32. Wang X, Ji H, Shi C, et al (2019) Heterogeneous graph attention network. In: WWW, https://doi.org/10.1145/3308558.3313562

33. Sun H, Li Y, Zhang Y (2022) ConLearn: Contextual-knowledge-aware Concept Prerequisite Relation Learning with Graph Neural Network, SIAM International Conference on Data Mining (SDM), pp 118–126. https://doi.org/10.1137/1.9781611977172.14, https://epubs.siam.org/doi/pdf/10.1137/1.9781611977172.14

34. Jia C, Shen Y, Tang Y, et al (2021) Heterogeneous graph neural networks for concept prerequisite relation learning in educational data. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Online, pp 2036–2047, https://doi.org/10.18653/v1/2021.naacl-main.164

35. Kipf TN, Welling M (2016) Variational graph auto-encoders. In: Proceedings of the 33rd International Conference on Machine Learning (ICML)

36. Li I, Fabbri AR, Tung RR, et al (2018) What should i learn first: Introducing lecturebank for nlp education and prerequisite chain learning. https://doi.org/10.48550/arXiv.1811.12181