A Study on the Construction of a Local Layered Knowledge Base for the Big Data Professional Curriculum System

Ying Li[1], Xiaodong Li[2,*], Wenjie Xiao[1], Lanping Zhang[1], Lilin Yang[1], Wenlin Zou[1], Yihan Shen[1]

[1] School of Information Engineering, NanJing XiaoZhuang University, Nanjing, China

[2] Machining Teaching and Research Section, Benxi Mechanical and Electrical Engineering School, Benxi, China

**Corresponding Author*

Abstract: To address the challenges posed by the rapid technological advancements, vast and complex knowledge structures, and heterogeneous data sources that hinder efficient information retrieval within the big data professional curriculum system, this study proposes an innovative approach to constructing a local layered knowledge base. Utilizing a hybrid retrieval weighted fusion algorithm, the method effectively normalizes and integrates disparate data. Experimental comparisons against a benchmark retrieval service based on a standardized general corpus demonstrate the superior effectiveness of the proposed method. Additionally, this approach offers robust data support aimed at enhancing employment-oriented course recommendations and facilitating personalized learning pathways for students.

Keywords: Local Layered Knowledge Base; Employment-Oriented; Vectorization; Knowledge Graph

1. Overview

The discipline of Big Data is intrinsically interdisciplinary, combining elements of computer science, statistics, mathematics, and specialized domain knowledge [1]. Its curriculum is both multifaceted and subject to rapid technological advancements. As a result, students frequently encounter difficulties when navigating employment-oriented course structures and accessing relevant learning resources. Existing university course information systems are insufficient in presenting a fully integrated and interconnected knowledge framework that aligns professional expertise with curriculum content in a coherent and comprehensive manner. While semantic knowledge graphs have improved the richness and completeness of information presentation, they often remain static and lack timely updates [2]. This shortcoming prevents students from obtaining comprehensive and current knowledge, thereby undermining their learning efficiency and restricting the realization of personalized learning pathways. In contrast, local knowledge bases built upon Retrieval-Augmented Generation (RAG) technology [3] enable swift and precise access to course details, academic planning guidance, and position recommendations through natural language interaction-without the need for complex signal feature extraction and modeling processes. Consequently, the effective construction and continuous real-time updating of such local knowledge base is crucial to supporting students' dynamic learning and career development needs.

In recent years, retrieval-augmented methods leveraging large language models have found extensive application across various specialized sectors such as education, healthcare, finance, and rail transit. However, most local knowledge bases predominantly rely on storing raw text embeddings within vector knowledge base, while some integrate both a vector knowledge base and a text knowledge base [4], and others employ knowledge graph-based constructions [5]. Although these approaches have somewhat enhanced retrieval efficiency, they typically follow a singular technological pathway, resulting in suboptimal collaborative retrieval performance across structured data, text, vectors, and graph-based information.

Given the vast and intricately structured knowledge system inherent to the big data professional curriculum domain, this paper innovatively proposes the construction of a layered local knowledge base. This architecture organizes domain-specific knowledge according to the intrinsic characteristics of the data sources, comprising a multi-tiered knowledge base encompassing a structured knowledge base, a text knowledge base, a vector knowledge base, and a knowledge graph. By integrating RAG models with advanced retrieval techniques, this framework achieves a structured consolidation and intelligent utilization of curriculum knowledge. It effectively transforms static materials into dynamic, interactive resources that facilitate course recommendations and personalized learning pathways. Ultimately, this approach provides data-driven support to pedagogical reform aimed at enhancing talent cultivation quality and employment competitiveness.

2. Solution Design

2.1 Overall Architecture

The architecture of the local layered knowledge base primarily comprises two integral components: knowledge processing and knowledge storage. Within the knowledge processing phase, diverse learning resources extracted from the big data professional curriculum-such as the curriculum system, project case studies, employment-oriented lecture videos, and competition entries [6]-are cleaned, deduplicated, and filtered for sensitive information from multiple sources. Leveraging a modular design alongside an array of advanced algorithms, the system processes this knowledge to construct four distinct knowledge bases: a structured knowledge base, a text knowledge base, a vector knowledge base, and a knowledge graph. This multifaceted knowledge base is engineered to accommodate heterogeneous query demands while significantly enhancing the speed and precision of information retrieval (illustrated in Figure 1).

The design of the local layered knowledge base adheres to a "data governance-layered knowledge transformation" framework, employing a modular architecture to achieve multimodal knowledge integration. Initially, a thorough needs analysis is performed on detailed data related to the professional training program, including course schedules, credit systems, syllabi, project cases, and other employment-oriented materials. This analysis culminates in the establishment of a four-tier knowledge model encompassing structured data, textual data, vector data, and knowledge graph data, as delineated in Table 1. During the data governance phase, heterogeneous multisource data are collected, and distinct knowledge base construction methodologies are applied according to the data type. Structured data such as course attributes, credits, and offering schedules undergo normalized modeling before storage in relational databases like PostgreSQL. Textual information, including course outlines, lecture notes, and academic papers, is processed via natural language processing techniques to extract ontological frameworks and coupled with Elasticsearch to facilitate semantically enhanced retrieval. Unstructured knowledge, such as project cases and ability atlases within the curriculum system, is vectorized using BERT embeddings and semantically matched through FAISS. Relationships involving course interrelations and position linkage—covering humanities literacy, technical proficiency, and professional competencies-are visualized through knowledge graphs constructed within Neo4j. By orchestrating relational databases, search engines, vector indices, and graph databases, the system builds a high-availability, interpretable multilayer local knowledge base, effectively supporting RAG systems with enhanced search capabilities.

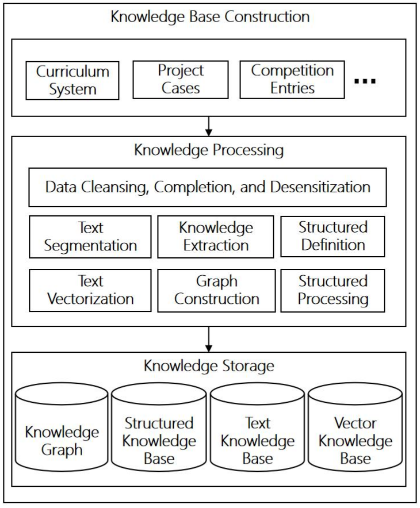

Figure 1. Architecture Diagram of the Local Knowledge Base

Table 1. Four-Tier Model of the Local Layered Knowledge Base

Type	Data Sources	Technical Implementation	Application Scenarios
Vector knowledge base	Curriculum Development, Competency Atlas	FAISS + BERT Embedding Models	Semantic Search, Intelligent Recommendation
Text knowledge	Course Outlines, Lecture	Elasticsearch	Full-text Search, Document

base	Notes, Papers		Association
Structured knowledge base	Course Schedules, Credits, Course Attributes	PostgreSQL	Course Query, Curriculum Optimization
Knowledge Graph	Course Relationships, Job Linkages	Neo4j	Academic Planning, Course Recommendation

2.2 Construction of the Vector Knowledge Base

The vector knowledge base serves as the pivotal engine for enabling semantic retrieval. Its fundamental function is to transform unstructured textual data-such as documents, paragraphs, and sentences-into vector representations, and subsequently identify the most semantically relevant information segments from vast datasets by computing the similarity between vectors. These retrieved fragments are then provided to large language models, empowering them to generate more accurate and contextually pertinent responses. This study employs the FAISS (Facebook AI Similarity Search) framework to implement the construction of the vector knowledge base [7].

2.2.1 Vectorization and data preprocessing

Within the big data professional curriculum ecosystem, there exists a plethora of multidimensional data encompassing employment statistics, course optimization metrics, weighting information, faculty details, and more. To enable these diverse datasets to be efficiently processed by the FAISS database, they must first undergo vectorization. This entails converting raw data—including textual content and numerical values—into high-dimensional vectors. For instance, employment-related data—such as geographic distribution of graduates, industry sectors, and salary ranges—can be encoded into feature vectors. Textual data, such as course descriptions or position requirement specifications, may be transformed into semantic vectors using pretrained models like BERT or Word2Vec. Through these vectorization techniques, originally heterogeneous data is rendered into a unified format amenable to mathematical manipulation and similarity computation.

2.2.2 Index construction in the FAISS database

Following the vectorization of data, the subsequent step involves the construction of the FAISS index. FAISS supports a variety of indexing methods, and within the context of curriculum system navigation and learning resource recommendation-where data volumes are substantial-commonly employed index structures include IVF (Inverted File) and HNSW (Hierarchical Navigable Small World).

(1) The IVF index partitions the dataset into multiple clusters, enabling queries to be executed only within relevant clusters. This drastically reduces the search space and significantly enhances query efficiency.

(2) The HNSW index, founded upon a small-world graph architecture, accelerates nearest neighbor search by constructing a graph structure, making it highly suitable for real-time retrieval over large-scale datasets.

Based on the characteristics of the data, an appropriate indexing strategy is selected. To further refine vectorization accuracy, a multi-granularity adaptive text segmentation algorithm is introduced. This approach integrates semantic boundary detection with syntactic dependency analysis to dynamically adjust the size of text segmentation windows, thereby mitigating the semantic fragmentation issues often caused by traditional fixed-size windows. By applying this algorithm, the text is partitioned into semantically coherent segments, more faithfully capturing the intrinsic meaning of the content and minimizing information loss.

2.3 Construction of the Text Knowledge Base

Traditional text retrieval systems typically employ highly efficient searching algorithms and indexing mechanisms, enabling the rapid extraction of the most relevant textual fragments from vast knowledge bases. This significantly accelerates the model's processing speed by preventing exhaustive, unguided searches through enormous datasets, thereby conserving both time and computational resources. The construction of the text knowledge base in this study leverages Elasticsearch’s inverted index mechanism, which fundamentally facilitates the swift retrieval of unstructured data through a document-to-term mapping approach. The inverted index consists of a lexicon and posting lists, following a reverse mapping logic from "term → document". Taking the core course "Operating Systems" within the big data professional curriculum as an exemplar, the pivotal stages of text knowledge base construction are as follows [8]:

2.3.1 Construction of the inverted index

(1) Term Generation: The IK tokenizer is used to perform maximum matching segmentation on text. For example, the phrase "the inter-process communication mechanism" is parsed into tokens such as (corresponding to "process", "inter-", "communication", "mechanism"). A stop-word filter is then used to remove semantically empty terms, including common grammatical particles like the preposition “the”.

(2) Index Structure: Each term is linked to a posting list containing document IDs, term frequencies (TF), and positions. For instance, for the term "deadlock"in Chapter 3 of the "Operating Systems" course—processor scheduling and deadlock—the posting can be represented as {docID: D045, TF: 3, positions: [12, 87, 153]}, indicating that "deadlock" appears three times within document segment number 045, specifically at the 12th, 87th, and 153rd positions within that segment.

(3) Compressed Storage: The dictionary is compressed using a Finite State Transducer (FST), encoding the lexicon into a minimized deterministic finite automaton. This technique drastically reduces the memory footprint of a lexicon comprising approximately 100,000 terms to merely 2.3 MB.

2.3.2 Semantic expansion mechanism

The synonym augmentation for the course "Operating Systems" is implemented through both static and dynamic expansion strategies:

(1) Static Expansion establishes preconfigured mapping rules between key technical terms and their designated equivalents. It defines direct correspondences such as "OS" to the concept of "Operating System" and "Mutex" to the synonym "Mutual Exclusion Lock".

(2) Dynamic Expansion employs co-occurrence analysis to automatically discover semantically related terms. The algorithm identifies pairs of words that frequently appear together within the same document or paragraph. When the co-occurrence frequency between two terms exceeds a predefined threshold (e.g., 50 co-occurrences), a bidirectional associative link is established between them and stored in an expansion dictionary. During query processing, the system uses this dictionary to replace or supplement original query terms with their associated counterparts, thereby broadening the search scope and improving recall. For instance, the system might learn that the term "process synchronization"is strongly associated with "thread synchronization" (co-occurrence score: 80) and "mutual exclusion access" (score: 60). Similarly, the term "mechanism" might be linked to "method" (score: 70) and "algorithm" (score: 65). Following the established associations, the query "process synchronization mechanism" would be algorithmically expanded to the following Boolean expression to enhance retrieval: (process synchronization OR thread synchronization OR mutual exclusion access) AND (mechanism OR method OR algorithm).

2.4 Construction of the Structured Knowledge Base

The structured knowledge base serves as a complementary counterpart to the vector knowledge base, addressing the latter’s limitations in handling precise matching, numerical computations, dynamic data updates, metadata filtering, and intricate relational queries. Structured data is stored and queried via relational databases, enabling exact matches and multi-criteria filtering through SQL. This structured knowledge base encompasses professional course information, enrollment requirements, faculty profiles, and more:

1) Course Information: This includes fundamental details such as course titles, credits, and instructional hours. These data points assist students in comprehending the basic attributes of each course while providing essential support for course administration and enrollment planning.

2) Enrollment Requirements: This section delineates both major-specific and interdisciplinary course pathways, credit prerequisites, and characteristics of practical courses. Such information underpins personalized learning plans tailored to individual student needs.

3) Faculty Profiles: Consisting of instructors’ research domains and courses taught, this aids students in selecting practical courses. Moreover, it facilitates alignment between faculty expertise and employment-oriented competition tracks for students.

Additionally, by integrating the Apriori algorithm, the system effectively uncovers associations between courses and positions, thereby augmenting course recommendation capabilities. Through meticulous data collection and preprocessing, the platform leverages Apriori to identify latent relationships between curriculum components and occupational skill requirements.

For instance, in analyzing the course "Spark Technology and Applications", the system detects "distributed computing" and "memory optimization" as its core competencies, assigning weights that reflect their significance for position roles. The implementation logic is exemplified below:

```
# Example: Association rules between "Spark Technology and Applications" and related competencies
if "Spark Principles" in course_content:
    add_relation(course, "distributed computing ability", weight=0.9)
    add_relation(course, "memory optimization ability", weight=0.7)
```

Through such a framework, the system generates an ability-demand graph for each course, thus empowering students in optimizing their course selections and facilitating curriculum refinement within their majors.

2.5 Construction of the Knowledge Graph

The Knowledge Graph (KG) assumes a pivotal role in advanced semantic comprehension and relational reasoning [9]. By structuring a network of interconnected entities, it remedies the shortcomings of conventional text-based and vector retrieval methods, particularly in logical inference, multi-hop queries, and dynamic knowledge association, thereby significantly enhancing the accuracy and interpretability of responses to complex inquiries. At the core of knowledge graph construction lies the modeling of relationships between entities—namely courses, abilities, and positions. The primary relationships encompass the "mapping between courses and abilities" as well as the "alignment between courses and position roles". Utilizing graph databases such as Neo4j for modeling, the system stores these relationships in graph form, enabling more flexible and efficient querying and inferencing [10, 11]. The ontology defines the fundamental concepts, entity types, and the interrelations among them within the knowledge graph.

1) Entity Types: Different classes of entities are defined, including course nodes, ability nodes, project nodes, etc. Each node carries multiple attributes to provide detailed contextual information, such as credits and instructional hours for courses, or ability levels and associated position roles for abilities.

Course (Course): Attributes include course title, category, credits, and instructional hours.

Knowledge Point (KnowledgePoint): Attributes include the knowledge point’s name and its domain.

Ability (Ability): Attributes encompass ability name, proficiency level, and relevant positions.

Project (Project): Attributes cover project name and project type.

2) Relationship Types: Upon defining the entity types, it is essential to specify the relationship types between entities by extracting connections from data sources such as course content, curriculum plans, and employment datasets. For instance, course nodes are linked to knowledge point nodes via a "CONTAINS" relationship, while knowledge points connect to ability nodes through a "DEVELOPS" relationship. The principal relationship types include:

BELONGS_TO: Course → Course Category
CONTAINS: Course → Knowledge Point
DEVELOPS: Knowledge Point → Ability
PREREQUISITE: Course → Course
RELATES_TO: Job Position → Project

Following the ontology definition, the data is stored within the Neo4j graph database, where each node (e.g., courses, abilities, positions) and each edge (such as "DEVELOPS" or "RELATES_TO") is represented as part of the graph structure. This graph-based representation elucidates the intricate interconnections among entities, thereby enabling highly efficient and sophisticated queries. Moreover, applying node embedding techniques such as GraphSAGE facilitates path reasoning along course dependency chains and ability requirements. The resultant knowledge graph thus serves as an intelligent backbone supporting tasks like course recommendation and academic planning.

3. Experiments and Result Analysis

3.1 Experimental Preparation

User queries initially pass through an intent recognition module, which subsequently routes them to the corresponding retrieval subsystems including the semantic vector search module, textual keyword search module, structured SQL query module, and a knowledge graph-based logical inference module. Ultimately, the results from these diverse sources are consolidated and prioritized via a fusion and ranking module, which constructs a prompt context fed into a large language model to generate the final response.

To achieve effective synergy among multi-source retrieval results, a hybrid retrieval weighted fusion algorithm is employed. This algorithm dynamically allocates weights to harmonize heterogeneous scores from the four retrieval sources into comparable values, as detailed in Equation (1):

$$final\_score = \alpha * S\_text + \beta * S\_vector + \gamma * S\_kg + \delta * S\_structured \quad (1)$$

Where, α, β, γ, and δ denote the weighting coefficients; S_text represents the textual retrieval score, S_vector corresponds to the cosine similarity from the vector knowledge base, S_kg indicates the path confidence within the knowledge graph, and S_structured stands for the numerical result of the structured knowledge base query. Given the differing scales of these parameters—for instance, BM25 scores in textual retrieval may exceed 100, whereas cosine similarity ranges between [-1,1]—normalization is requisite to map all metrics into the [0,1] interval (e.g., S_vector = (cos_sim + 1)/2), thus ensuring equitable weighting during score amalgamation.

3.2 Experimental Results Analysis

To validate the effectiveness of the proposed local layered knowledge base, this study devised a multi-channel RAG-enhanced retrieval experimental framework. The evaluation primarily focused on retrieval efficiency, the relevance of generated content, and the precision in supporting employment-oriented queries.

3.2.1 Dataset

A comprehensive dataset was curated, reflecting authentic scenarios within the talent cultivation pipeline of the Big Data discipline, encompassing courses, projects, competitions, and employment facets. The dataset components include:

(1) Structured Data: Comprising approximately 500 entries sourced from university academic administration systems, including course schedules, credit frameworks, and faculty profiles.

(2) Textual Data: Encompassing roughly 1,000 documents derived from course materials, technical lectures, and corporate recruitment notices.

(3) Vector Data: Generated by embedding course descriptions, ability requirements, and related content using FAISS and BERT models, resulting in approximately 1,000 course vectors.

(4) Knowledge Graph Data: Constructed based on relationships among courses, abilities, and positions, this graph contains about 500 nodes spanning these entities.

3.2.2 Experimental results analysis

The experiment contrasted the benchmark retrieval method based on a standard general corpus retrieval service against the multi-channel RAG-enhanced retrieval approach founded on the proposed local layered knowledge base.

(1) Retrieval Efficiency Comparison

Regarding retrieval efficiency, we evaluated the response times and recall rates across different retrieval modules. The results, summarized in Table 2, report the average response times for SQL-based structured knowledge base retrieval, Elasticsearch-based textual retrieval, FAISS-based vector retrieval, and Neo4j-driven knowledge graph retrieval. The findings demonstrate that the method leveraging the local layered knowledge base markedly outperforms the benchmark approach in retrieval efficiency.

Table 2. Comparison of Retrieval Efficiency

Retrieval Module	Proposed Method (ms)	Benchmark Method (ms)	Improvement (%)
Vector knowledge base	100	120	16%
Text knowledge base	98	150	34%
Structured knowledge base	65	100	35%
Knowledge Graph	120	180	33%

(2) Relevance of Generated Content

To assess the relevance of the generated content, both manual and automated evaluation methods were employed. The manual assessment was conducted by three experts who rated the outputs based on relevance, accuracy, and practicality with respect to the queries, using a scale from 1 to 5, where 5 denotes the highest score. The automated evaluation utilized the BLEU (Bilingual Evaluation Understudy) metric to quantitatively measure the quality of the generated text. The results are presented in Table 3.

Table 3. Comparison of Relevance of Generated Content

Method	Average Manual Score	BLEU Score
Proposed Method	4.8	0.89
Benchmark Method	3.5	0.74

The comparative analysis of manual scores and BLEU ratings reveals that the multi-channel RAG-enhanced retrieval method based on the local layered knowledge base significantly improves the relevance of generated content, with an approximate 37% enhancement in content pertinence.

(3) Accuracy of Employment-Oriented Support

We further evaluated the accuracy of the knowledge base in providing employment-oriented course recommendations and learning plan guidance. Thirty representative query scenarios were designed, encompassing course selection, course recommendation, learning pathway design, and career development advice. Experts rated each recommendation based on the degree of personalization and accuracy. (As shown in table 4)

Table 4. Comparison of Supported Accuracy

Method	Recommendation (%)	Personalization Score (1–5)
Proposed Method	86	4.7
Benchmark Method	76	3.9

The experimental results demonstrate that the local layered knowledge base approach excels in both the precision and personalization of academic cultivation and career guidance, achieving a 10% improvement in recommendation accuracy over the benchmark method.

4. Conclusion

The local layered knowledge base proposed for the big data professional curriculum system demonstrates significant advantages in retrieval efficiency, relevance of generated content, and accuracy of employment-oriented support within the context of big data course knowledge retrieval. The experimental outcomes substantiate the method's promising potential to enhance student course recommendation and personalized learning path design driven by career needs. Nevertheless, with continual technological advancement and evolving application demands, there remains room for refinement. Particularly, future research must address the critical challenge of more effectively integrating heterogeneous educational data types, thereby elevating semantic coherence of information and enhancing precision in retrieval within the complex domain of education.

References

[1] Yang Bo, Li Yuanbiao. Complex Network Analysis on Curriculum System of Data Science and Big Data Technology. Computer Science, 2022, 49(S1): 680-685+807.

[2] Yu Chengcheng, Shi Linxiang, Chen Lin, et al. Construction and Exploration of "Big Data Technology" AI Course Based on Knowledge Graph. Computer Era, 2025(10): 95-99.

[3] Patrick Lewis, Ethan Perez, Aleksandara Piktus, et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems, 2020,33, 9459-9474.

[4] Liu Zhenyi. Research on Construction and Application of Computer Course Knowledge Base Based on RAG. Computer Knowledge and Technology, 2025, 21(08): 26-28.

[5] Yu Songwei, Liu Wei, Xia Xiujiang, et al. Constructing a Retrieval-Augmented Generation Knowledge Base for Urban Rail Transit Large Language Models: A Knowledge Graph-Based Approach. Urban Rapid Rail Transit, 2025, 38(02): 19-25.

[6] Hou Pengliang, Zhang Fulong, Xiao Haining, et al. An Exploration of Teaching Reform to Promote Learning by Competitions and Promote Teaching by Competitions. Journal of Educational Institute of Jilin Province, 2024, 40(1): 121-126.

[7] Shang Xueru, Chen Han. Optimal Conceptual Semantic Template Self Retrieval Based on Structured Corpus. Computer Simulation, 2025, 42(01): 550-554.

[8] Wang Xiaoling, Yue Wenjing, Wang Haofen, et al. Teaching Exploration of Integrating Large Language Model Technology into Database Courses. Computer Education, 2024(9): 28-32.

[9] Zhao Yubo, Zhang Liping, Yan Sheng, Hou Min, Gao Mao. Construction and Application of Discipline Knowledge Graph in Personalized Learning. Computer Engineering and Applications, 2023, 59(10): 1-21.

[10] Huang Qiaojuan, Cao Cungen, Wang Ya, et al. Method for Expanding Event Commonsense Knowledge Graph Based on Large Language Models. Journal of Software, 2025, 36(09): 4153-4186.

[11] Song Meixia, Zhang Shuaishuai. The Essence, Reality and Optimization Path of ChatGPT-Enabled Personalized Learning. Journal of Continuing Higher Education, 2023, 36(5): 73-80.