Graph-Based Personalized Recommendation in Intelligent Educational Platforms: A Case Study in Engineering Education

Sofia Merino Costa[1], Rui Pinto[2][a] and Gil Gonçalves[2][b]
[1]*Departamento Engenharia Informática, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal*
[2]*SYSTEC-ARISE, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal*

Keywords: Graph-Based Recommendation, Knowledge Management System, Education 5.0, Engineering Education.

Abstract: The fragmentation of digital learning materials in engineering education makes it difficult for students to find relevant content. This paper presents a graph-based recommender system integrated into an intelligent Knowledge Management System (KMS) to support personalized learning. Using Neo4j, the system models users, learning objects, and semantic relationships to generate contextualized recommendations across dashboard, module, and Learning Path (LP) views. Its scoring mechanism combines semantic similarity, interaction history, and graph proximity to provide adaptive, explainable suggestions. A mixed-methods evaluation with engineering students showed high alignment with user interests and positive perceptions of transparency and personalization. The system effectively transitioned from fallback to tailored recommendations as user interactions increased. Results highlight the potential of graph-based approaches to improve content relevance, discovery, and learner engagement in web-based educational platforms, in line with Education 5.0 principles.

1 INTRODUCTION

Engineering students increasingly face the challenge of navigating vast and complex digital learning ecosystems. While abundant resources, such as lecture notes, academic papers, and interactive tools, offer rich learning opportunities, they also contribute to information overload, making it difficult for learners to identify relevant, trustworthy, and pedagogically aligned content (Dalkir, 2017). Traditional platforms often provide limited support for this discovery process, relying on static listings, keyword-based search, and non-personalized navigation (Kopeyev et al., 2020).

To address this, there is growing interest in integrating intelligent features into web-based educational environments. Among these, recommender systems offer promise by matching content to learners based on usage patterns, metadata, and learning objectives (G. M. et al., 2024). However, their application in education remains limited, particularly in terms of explainability, semantic richness, and adaptability to evolving learner needs (Sahu et al., 2024; Liu et al., 2024a).

The main contributions of this paper are as follows:

- A graph-based recommendation approach for an intelligent Knowledge Management System (KMS), tailored for educational resource discovery using *Neo4j* (Webber and Robinson, 2018) to model learners, content, and pedagogical relationships.
- Multi-context recommendation logic tailored to user states (e.g., new, returning, active) and interface views (dashboard, module, learning path).
- A mixed-methods evaluation combining precision metrics and user feedback to assess recommendation quality and perceived usefulness.
- A discussion on how explainability and contextual relevance foster learner trust and engagement in alignment with Education 5.0.

The remainder of this paper is structured as follows: Section 2 reviews related work on educational recommender systems and graph-based approaches. Section 3 presents the system architecture and recommendation logic. Section 4 details the evaluation methodology and results. Section 5 discusses the findings, limitations, and implications. Finally, Section 6 concludes the paper and outlines directions for future work.

[a] https://orcid.org/0000-0002-0345-1208
[b] https://orcid.org/0000-0001-7757-7308

Costa, S. M., Pinto, R. and Gonçalves, G.
Graph-Based Personalized Recommendation in Intelligent Educational Platforms: A Case Study in Engineering Education.
DOI: 10.5220/0013830200003985

In *Proceedings of the 21st International Conference on Web Information Systems and Technologies (WEBIST 2025)*, pages 439-446
ISBN: 978-989-758-772-6; ISSN: 2184-3252

2 RELATED WORK

Recommender systems are central to intelligent educational platforms, offering personalized content delivery and helping students navigate large volumes of heterogeneous learning materials. In engineering education, this need is particularly acute, as students must identify relevant resources across complex, interdisciplinary domains, often without structured guidance (Urdaneta-Ponte et al., 2021).

Traditional recommendation techniques fall into two main categories: collaborative filtering, which identifies patterns based on user-item interactions, and content-based filtering, which relies on semantic features and metadata (Lops et al., 2011; Ricci et al., 2011). While effective in domains such as e-commerce, these approaches face limitations in education due to sparse behavioral data, misalignment with pedagogical goals, and the cold-start problem (Burke, 2002).

To address these challenges, hybrid systems have emerged, combining collaborative and content-based methods. A significant advancement is the use of graph-based representations, where users, resources, and educational concepts are modeled as interconnected nodes. This structure enables semantic enrichment, multi-hop reasoning, and explainable recommendations—essential features for transparency and trust in learning environments (Markchom et al., 2023).

Recent research reflects this shift toward graph-enhanced educational recommenders. (Lu and Feng, 2024) combined graph convolutional networks with user behavior modeling for dynamic adaptation. (Liu et al., 2024b) used *Neo4j* to model user-resource relationships, integrating NLP to enhance semantic query interpretation(Webber and Robinson, 2018). (Hu et al., 2021) addressed information overload by filtering resources based on rule-based matching and user profiles. (Imamah et al., 2024) applied Ant Colony Optimization to personalize Learning Paths (LPs) based on difficulty and learner preferences. Meanwhile, Skillify (Dhairya et al., 2024) integrated generative AI features to recommend content based on learner skill gaps.

These contributions highlight the pedagogical value of recommender systems, particularly when enhanced by graph structures, semantic reasoning, and adaptive logic. However, key limitations remain:

- Many systems struggle with cold-start scenarios and sparse data.
- Few support multi-context recommendations aligned with user state and platform usage.
- Explainability, though feasible via graph-based logic, is underutilized.
- Alignment with Education 5.0 values—human-centricity, personalization, and intelligent support—remains limited.

This work addresses these gaps by introducing a modular graph-based recommender embedded in an educational KMS for engineering students. The system provides context-aware recommendations across dashboard, module, and LP views, leveraging semantic links for pedagogical relevance. Its graph-based foundation also enables future enhancements in explainability, learner modeling, and adaptive reasoning, supporting Education 5.0 principles (Pinto et al., 2023; Pinto et al., 2024).

3 SYSTEM DESIGN

The platform developed in this work serves as a prototype for an intelligent KMS designed to enhance engineering education. The system is composed of four main components: I) a frontend interface for user interaction; II) a backend server that handles application logic, data management, and API communication; III) a data layer including a relational database and a graph-based recommendation system; and IV) external services such as a semantic search engine and cloud storage. This modular architecture, as represented in Figure 1, ensures a seamless user experience and provides scalability for future growth. Although the system is not fully adaptive or self-learning, it integrates foundational intelligent features such as semantic information retrieval and graph-based personalization, which represent an important step toward creating more advanced, intelligent systems.

The recommendation system is a core component of the broader intelligent knowledge management platform tailored for engineering students. This platform supports modular content organization, semantic search, LPs, and personalized recommendations to guide students through relevant learning materials efficiently. To meet these objectives, the architecture adopts a hybrid multi-database backend, combining *PostgreSQL* (Obe and Hsu, 2017) for relational content and user management, *Elasticsearch* (Konda, 2024) for semantic retrieval, and *Neo4j* (Webber and Robinson, 2018) (Community Edition) as a graph-based recommendation engine. Unlike triple-store Resource Description Frameworks (RDF), this approach was designed as a database model (rather than a data exchange format), and easily handles multiple relationships of the same type between the same two nodes.

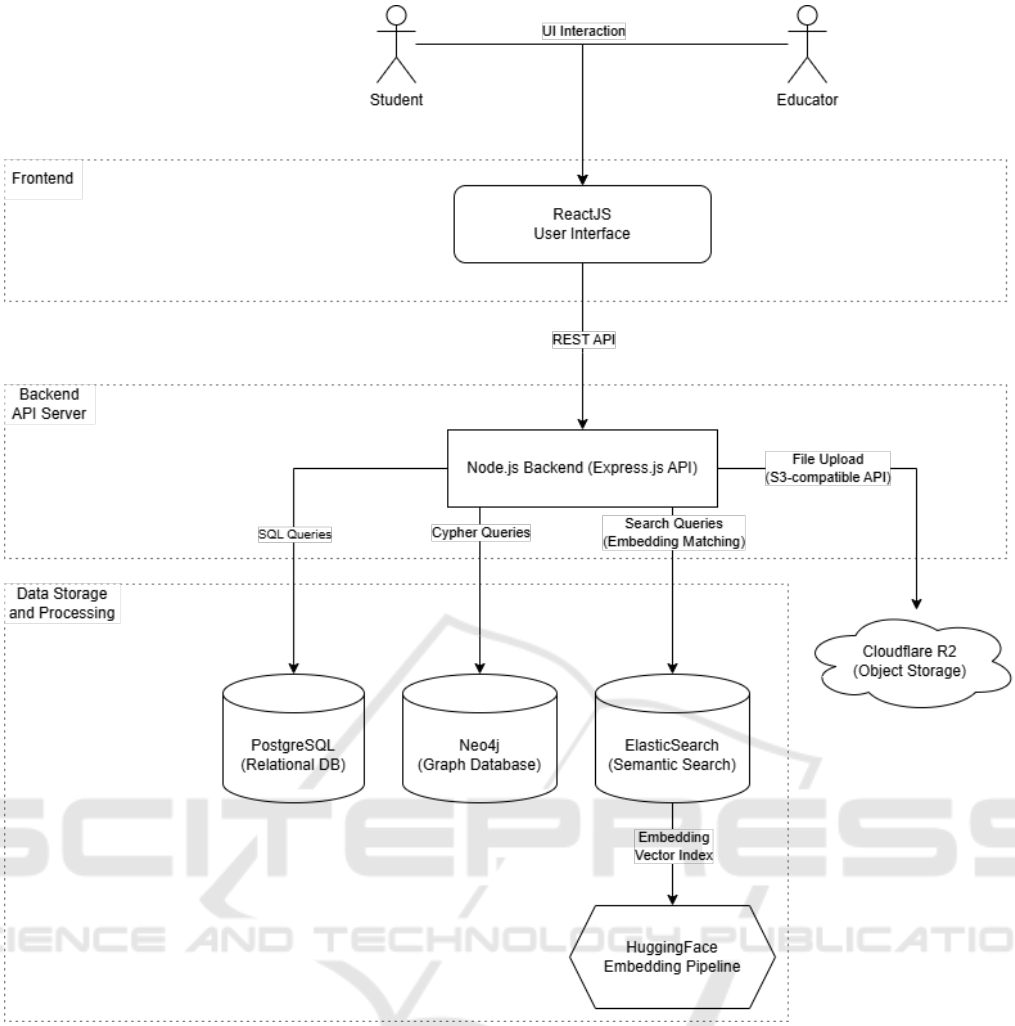

Figure 1: Component and Data Flow Diagram.

3.1 Data Model & Synchronization

The platform's ability to support intelligent features such as personalized recommendations, semantic search, and adaptive LPs relies heavily on a robust and well-structured data model. The choice to combine relational, search, and graph databases stems from their complementary strengths. First, *PostgreSQL* ensures reliable storage and transactional consistency of structured data such as user profiles, content metadata, and LP structures. Second, *Elasticsearch* enables fast and flexible semantic search across textual resource descriptions, supporting rich query capabilities beyond keyword matching. Finally, *Neo4j*, a property graph database, powers the recommendation engine by efficiently modeling complex, evolving relationships between users, learning resources, modules, and LPs, enabling context-aware traversal queries that adapt to learner behaviors.

To integrate data from the relational backend into the graph, a custom synchronization pipeline is implemented via a *Node.js* script (Mardan et al., 2018). This script extracts relevant entities and relationships from *PostgreSQL* and incrementally populates the *Neo4j* graph. It clears existing data and recreates nodes for users, resources, modules, LPs, and classification nodes such as categories and tags, establishing their interconnections accordingly. User interactions, such as resource views, module starts, bookmarks, and completions, are stored relationally and then represented as explicit `Interaction` nodes in the graph, linked to the corresponding `User` and `Resource`, `Module`, or `LearningPath` targets. This design enables rich traversal patterns and weighted personalization.

3.1.1 Neo4j Graph Schema

To support personalized recommendations, the platform incorporates a graph-based model implemented with *Neo4j*. This structure enables efficient traversal of interconnected entities, including users, resources, modules, and LPs, capturing complex relationships not easily expressed in relational databases (Figure 2). Key educational elements are modeled as nodes, such as:

- `User`: Learners with profile attributes (e.g., education level, field of study, topic interests, preferred content types, language preferences).
- `Resource`: Learning materials enriched with metadata such as categories, tags, type, title, and description.
- `Module`: Groupings of related resources representing standalone instructional units.
- `Learning Path`: Ordered sequences of modules guiding structured progression.
- `Category`, `Tag`, `ResourceType`: Classification nodes supporting semantic connections.
- `Interaction`: User actions (e.g., view, start, bookmark, complete) linked to targets, with timestamps and weights.

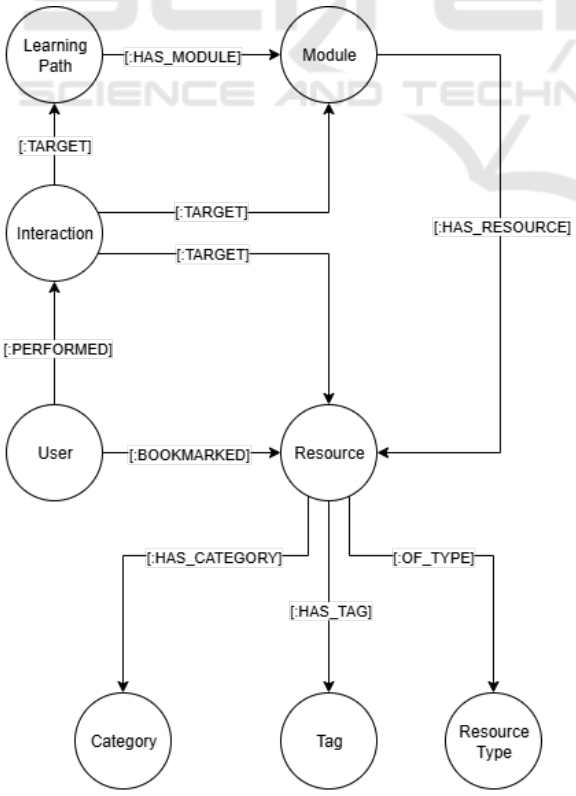

Figure 2: Graph-based recommendation schema.

Relationships encode these connections, such as `(:Module)-[:HAS_RESOURCE]->(:Resource)` and `(:LearningPath)-[:HAS_MODULE]->(:Module)` for content hierarchy, or `(:User)-[:PERFORMED]->(:Interaction)` and `(:Interaction)-[:TARGET]->(:Resource)` for tracking user behavior. Each `Interaction` node contains properties such as interaction type, timestamp, and a computed weight.

The graph structure allows the system to perform personalized content recommendations based on traversals of relationships such as shared tags, co-interacted resources, or similar LPs. *Cypher* queries (He et al., 2022) are used to retrieve relevant content for each user based on both direct and indirect connections, enabling a more contextualized and intelligent experience.

3.2 Recommendation and Scoring Logic

The recommendation engine integrates three core dimensions: Content Similarity (*CS*), Interaction Weighting (*IW*), and Graph Proximity (*GP*). When a learner interacts with a resource, module, or LP, the system traverses the graph to identify candidate content and computes a final recommendation score using a weighted sum, as represented in Equation 1:

$$\text{Score} = w_1 \times CS + w_2 \times IW + w_3 \times GP \quad (1)$$

The weights w_1, w_2, and w_3 are empirically tuned to balance semantic, behavioral, and structural signals.

Content Similarity (*CS*) captures thematic closeness by comparing tags, categories, and metadata. The system employs the *Sørensen–Dice* coefficient (Gragera and Suppakitpaisarn, 2016), implemented via *Neo4j*'s Awesome Procedures on Cypher (APOC) plugin (Shatnawi and Saquer, 2024), to compute semantic similarity between resources.

Interaction Weighting (*IW*) reflects the pedagogical significance and recency of user actions. Each interaction type is assigned a base weight (w_{base}), adjusted by a linear decay function that prioritizes recent activity, as represented in Equation 2:

$$w_x = w_{base} \times \max(1, 10 - DaysSinceInteraction) \quad (2)$$

The interaction types and corresponding w_{base} values are: 1 - Default (other interactions); 2 - Resource/module/LP views; 3 - Module/LP started; 4 - Content bookmarked; 5 - Module completed; 6 - LP completed.

This approach ensures that recent, pedagogically meaningful actions have greater influence on recommendations, while older actions still contribute with diminished weight.

Graph Proximity (*GP*) encodes the structural relationship between users and content. Shorter path lengths in the graph indicate stronger contextual relevance, enabling the system to blend content-based and collaborative filtering perspectives.

Together, these three dimensions enable the system to deliver adaptive, context-aware, and explainable recommendations aligned with each learner's evolving profile and engagement history.

3.3 Recommendation Query Formulation

The recommendation engine generates context-specific suggestions using tailored *Cypher* queries executed on the underlying graph. These queries are designed to balance personalization, semantic relevance, and robustness to sparse data. Core elements of the query formulation include:

- **Interaction-Based Personalization:** When available, the engine prioritizes user interaction history, excluding previously accessed content and surfacing items aligned with past engagement patterns and preferences.
- **Profile-Driven Filtering:** In the absence of interaction data (e.g., cold-start), recommendations are derived from explicit profile attributes such as topic interests and preferred content formats.
- **Semantic Matching:** Queries incorporate tag similarity, category alignment, and metadata comparisons to surface semantically related content within the current context (e.g., a viewed module or resource).
- **Popularity Signals:** Aggregate usage data across the platform is used to highlight frequently accessed or highly rated content, improving diversity and baseline relevance.
- **Graph-Based Contextualization:** The engine exploits both direct and indirect relationships within the graph—such as module-resource links or shared attributes across LPs—to infer pedagogical and thematic connections.
- **Multi-Tier Fallbacks:** A hierarchical fallback strategy ensures continuity in recommendations, progressing from personalized to profile-based and finally to globally popular content as needed.

These dynamic query patterns leverage *Neo4j*'s aggregation capabilities to compute ranked recommendations across key platform views, including the dashboard, resource detail pages, and LP modules.

3.4 Recommendation Contexts and Pedagogical Impact

The recommendation engine operates across multiple contexts within the platform, each aligned with a specific stage of the learner's journey:

- **Dashboard:** Upon login, students receive personalized recommendations for resources, modules, and LPs based on their profile, declared interests, and recent interactions. This reduces the cognitive load of content discovery and promotes re-engagement.
- **Resource View:** When accessing a specific resource, the system suggests semantically related materials, encouraging exploratory learning and thematic deepening.
- **Module View:** Within a module, recommendations highlight complementary modules or LPs that support conceptual progression and curriculum continuity.
- **Learning Path View:** For learners navigating a structured LP, the system offers reinforcing resources and related modules to strengthen understanding and align with learning objectives.

Each context employs a tailored version of the scoring logic, ensuring that recommendations are both contextually relevant and pedagogically aligned.

4 EVALUATION

This section presents an evaluation of the intelligent KMS developed in this work, with a focus on assessing the performance of its personalized recommendations component. The evaluation was designed to reflect realistic usage within the context of engineering education, by testing the system across all major contexts where recommendations appear within the platform, as represented in Table 1.

Table 1: Recommendation contexts across the platform.

Context	Description
Dashboard	Personalized suggestions based on profile and interaction data.
Resource Page	Related resources and modules extending the current topic.
Module Page	Additional modules and LPs supporting progression.
LP Page	Resources that deepen knowledge in active LPs.

Each context was tested under multiple user states, namely: I) cold start, i.e., new user with no data

regarding preferences (content topics and type); II) profile-based (static preferences selected by a new or returning user); and III) personalized (interaction-aware, after multiple interactions of a returning user), to evaluate system behavior in both ideal and constrained conditions. These controlled conditions allowed for consistent comparison of fallback and personalized recommendation strategies.

Ten participants, students from computer science programs, completed a structured sequence of 3 tasks designed to simulate the system's intended use. Also, the system wasn't used in real courses. The tasks are: I) Participants registered into the system, selected their preferences, and configured their profiles. This task was used to test the cold-start logic of the recommendation engine and capture profile-based personalization; II) The new users were asked to interact with their personalized dashboard. They were encouraged to bookmark useful resources, interact with recommendations, and rate their relevance based on the submitted preferences at sign-up. This task was used to test the performance of the *Neo4j*-based graph recommendation system and assess the perceived quality of personalization; III) Participants selected one of two LPs and completed its modules. Each module included resources and an assessment to verify knowledge gained.

The system was evaluated using the standard quantitative metric Precision@k (P@k), which measures the proportion of relevant items among the top k recommendations, calculated using Equation 3. This metric was selected due to its widespread use and suitability for top-k recommendation tasks. High precision indicates strong immediate relevance.

$$\text{P@}k = \frac{\#\text{ relevant items in top } k}{k} \tag{3}$$

4.1 Results Summary

The following presents the system's performance across the different recommendation scenarios identified in Table 1. Metrics are reported for each context to reflect how well the system adapts to different user states and content configurations.

4.1.1 Dashboard

Table 2 shows the results for the dashboard context, where user-level personalization is expected to be most impactful. The system showed strong performance in profile-based and personalized scenarios, i.e., achieving a perfect P@k of 1.00. As expected, no metrics are reported for cold start users, where fallback to popular items was applied instead of personalized ranking. These results demonstrate the value of user modeling: when preferences or interaction data are available, highly relevant recommendations are consistently retrieved.

Table 2: Dashboard recommendation relevance by user state.

State	Fallback	P@k
Cold Start	Popular	–
Profile-Based	Preferences	1.00
Personalized	N/A	1.00

4.1.2 Resource Page

Table 3 presents the results for module recommendations on resource pages.

Table 3: Resource page module recommendation results.

Resource Type	P@k	Fallback
With Module	1.00	No
Standalone	1.00	Yes

Even when fallback logic was needed (for standalone resources), the system maintained high relevance, achieving perfect scores in both contexts. These results suggest that the semantic similarity logic and graph-based expansion strategies used for recommending adjacent modules are effective even in the absence of direct contextual anchors.

4.1.3 Module Page

Table 4 shows how well the system suggested LPs when users viewed modules.

Table 4: Module page LP recommendation relevance.

Module Type	P@k	Fallback
With Path	1.00	No
Without Path	0.67	Yes

P@k dropped to 0.67 for standalone modules due to fallback recommendations, highlighting some gaps in topic alignment. These findings suggest a need for improved fallback mechanisms in cases where modules are not yet connected to curated LPs.

4.1.4 Learning Path Page

Table 5 summarizes the relevance of resources recommended on LP pages. The three evaluated LPs were: *Introduction to Knowledge Management Systems* (LP1), *Getting Started with Arduino* (LP2), and *Cyber-Physical Systems and Internet of Things* (LP3).

While the system returned relevant items for all LPs, performance varied considerably. Notably, LP1 and LP2 exhibited no unique recommendations, with

Table 5: LP page resource recommendation results.

LP	Items	Unique	Relevant	P@k
LP1	6	0	3	0.50
LP2	6	0	2	0.33
LP3	6	2	4	0.67

multiple duplicate items across paths, resulting in lower P@k scores of 0.50 and 0.33 respectively. In contrast, LP3, which contained more distinct and well-tagged content, achieved better diversity with two unique recommendations and a higher P@k of 0.67. This variation indicates that while the system can surface relevant content, there is room to improve its ability to balance relevance and novelty, particularly when multiple paths are thematically similar.

5 DISCUSSION

The evaluation highlighted both the strengths and limitations of the recommendation system, showing how performance is shaped by user context, interaction history, and the structure of the underlying content graph. In personalized scenarios, particularly the dashboard view, recommendations consistently achieved perfect P@k scores when user profile data and interaction history were available. This validates the scoring strategy, which combines semantic similarity, weighted user interactions, and graph proximity to produce contextually relevant suggestions. The graph-based design, together with scoring logic, allowed dynamic adaptation to evolving user states, transitioning smoothly from fallback suggestions to personalized results. Users positively noted this progression, perceiving a clear improvement in recommendation quality over the course of their session.

Despite these strengths, several limitations were observed. In cold-start scenarios, fallback recommendations relied on content popularity. While this ensured a baseline level of relevance, it often failed to align with users' specific interests. Additionally, in the LP detail view, content overlap was observed between LPs that addressed different topics. This issue, particularly between LP1 and LP2, with P@6 scores of 0.50 and 0.33, reflected insufficient differentiation in resource tagging and semantic descriptors. LP3, featuring more distinctive and well-annotated content, performed better (P@6 = 0.67), reinforcing the importance of metadata richness.

These findings suggest that recommendation quality is not solely dependent on algorithmic logic, but also on the breadth, granularity, and semantic quality of the content graph. Sparse metadata and limited content variety constrain the system's ability to generate diverse or specialized suggestions. Enhancing semantic modeling—via knowledge graph embeddings or NLP-based descriptors—could improve thematic sensitivity and recommendation diversity. Moreover, although the system's graph structure inherently supports explainability, the lack of a user-facing explanation layer limits its impact on trust and transparency, which are critical in educational settings.

In summary, the system's strong performance in personalized contexts supports the validity of its graph-based architecture and scoring logic. However, the modest results under cold-start and semantically sparse conditions point to areas for future work, including improved metadata enrichment, integration of explainable interfaces, and scalable strategies for expanding and maintaining the content graph. These enhancements are key to deploying intelligent, learner-centered recommendation systems in real-world educational environments.

6 CONCLUSION & FUTURE WORK

This paper presented a graph-based recommender system integrated into an intelligent KMS for engineering education. Aligned with Education 5.0 principles, the system delivers personalized, context-aware recommendations across multiple learning views—including the dashboard, module pages, and learning paths. Leveraging *Neo4j* to model users, content, and semantic relationships, the system dynamically adapts to learner profiles and activity states, using a hybrid scoring strategy to provide relevant suggestions—even during early-stage interactions.

Evaluation results confirmed the system's effectiveness in personalized contexts, validating the multi-factor scoring logic that combines semantic similarity, interaction history, and graph proximity. Users experienced a clear progression from fallback to personalized recommendations during sessions. However, reduced performance in cold-start and semantically sparse scenarios underscored the need for richer metadata, broader content coverage, and enhanced similarity modeling.

This work contributes a practical, explainable, and modular recommendation framework for educational platforms, balancing adaptability with integration flexibility. Future development will focus on expanding the semantic layer, incorporating real-time behavioral signals, and implementing visual explanation features to enhance transparency, learner trust, and long-term engagement in self-directed learning.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support provided by the Foundation for Science and Technology (FCT/MCTES) within the scope of the Associated Laboratory ARISE (LA/P/0112/2020), the R&D Unit SYSTEC through Base (UIDB/00147/2020) and Programmatic (UIDP/00147/2020) funds

REFERENCES

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370.

Dalkir, K. (2017). *Knowledge Management in Theory and Practice*. MIT Press, 3rd edition.

Dhairya, Hrishikesh, J., Sonu, G. P., Vaishnavi, D., Sanapala, S., and Pallavi, L. (2024). Skillify: Enhanced Learning Management System Using Generative AI. In *2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT)*, volume 1, pages 1527–1532.

G. M., D., Goudar, R. H., Kulkarni, A. A., Rathod, V. N., and Hukkeri, G. S. (2024). A digital recommendation system for personalized learning to enhance online education: A review. *IEEE Access*, 12:34019–34041.

Gragera, A. and Suppakitpaisarn, V. (2016). Semimetric properties of sørensen-dice and tversky indexes. In Kaykobad, M. and Petreschi, R., editors, *WALCOM: Algorithms and Computation*, pages 339–350, Cham. Springer International Publishing.

He, Z., Yu, J., and Guo, B. (2022). Execution time prediction for cypher queries in the neo4j database using a learning approach. *Symmetry*, 14(1).

Hu, H., Ma, X., Pan, H., Zhang, H., and Gu, F. (2021). Research on Intelligent Recommendation System Based on Knowledge Management. In *2021 International Conference on Intelligent Computing, Automation and Applications (ICAA)*, pages 195–198.

Imamah, Laili Yuhana, U., Djunaidy, A., and Purnomo, M. H. (2024). Development of Dynamic Personalized Learning Paths Based on Knowledge Preferences and the Ant Colony Algorithm. *IEEE Access*, 12:144193–144207. Conference Name: IEEE Access.

Konda, M. (2024). *Elasticsearch in action*. Simon and Schuster.

Kopeyev, Z., Mubarakov, A., Kultan, J., Aimicheva, G., and Tuyakov, Y. (2020). Using a personalized learning style and google classroom technology to bridge the knowledge gap on computer science. *International Journal of Emerging Technologies in Learning (IJET)*, 15(2):218–229.

Liu, J., Luan, M., and Jiang, N. (2024a). Personalized academic communication recommendation system based on knowledge graph. In *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, pages 610–614.

Liu, J., Luan, M., and Jiang, N. (2024b). Personalized Academic Communication Recommendation System Based on Knowledge Graph. In *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, pages 610–614.

Lops, P., Gemmis, M. d., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. Springer.

Lu, P. and Feng, X. (2024). Personalized Recommendation Algorithm in AI-Assisted Learning System. In *2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 1289–1294. ISSN: 2833-2423.

Mardan, A., Mardan, and Corrigan (2018). *Practical Node. js*. Springer.

Markchom, T., Liang, H., and Ferryman, J. (2023). Review of explainable graph-based recommender systems. *ACM Computing Surveys*.

Obe, R. O. and Hsu, L. S. (2017). *PostgreSQL: up and running: a practical guide to the advanced open source database*. " O'Reilly Media, Inc.".

Pinto, R., Pinheiro, J., Gonçalves, G., and Ribeiro, A. (2023). Towards industry 5.0: A capacitation approach for upskilling and technology transfer. In Mehmood, R., Alves, V., Praça, I., Wikarek, J., Parra-Domínguez, J., Loukanova, R., de Miguel, I., Pinto, T., Nunes, R., and Ricca, M., editors, *Distributed Computing and Artificial Intelligence, Special Sessions I, 20th International Conference*, pages 342–351, Cham. Springer Nature Switzerland.

Pinto, R., Žilka, M., Zanoli, T., Kolesnikov, M. V., and Gonçalves, G. (2024). Enabling professionals for industry 5.0: The self-made programme. In *5th International Conference on Industry 4.0 and Smart Manufacturing (ISM 2023)*, volume 232, pages 2911–2920.

Ricci, F., Rokach, L., and Shapira, B. (2011). *Introduction to recommender systems handbook*, pages 1–35. Springer.

Sahu, S. S., Kumar, R., Sahoo, S., Kumar, B., and Mohanta, P. (2024). *Machine Learning-Enabled Integrated Information Platform for Educational Universities*, chapter 2, pages 29–47. John Wiley & Sons, Ltd.

Shatnawi, H. and Saquer, J. (2024). Encoding feature models in neo4j graph database. In *Proceedings of the 2024 ACM Southeast Conference*, ACMSE '24, page 157–166, New York, NY, USA. Association for Computing Machinery.

Urdaneta-Ponte, M. C., Mendez-Zorrilla, A., and Oleagordia-Ruiz, I. (2021). Recommendation systems for education: Systematic review. *Electronics*, 10(14).

Webber, J. and Robinson, I. (2018). *A Programmatic Introduction to Neo4j*. Addison-Wesley Professional, 1st edition.