Journal of Universal Computer Science, vol. 30, no. 12 (2024), 1645-1661
submitted: 1/11/2023, accepted: 16/5/2024, appeared: 28/11/2024

Using Adaptive Content Recommendations to Improve Logic and Programming Teaching and Learning

Aluizio Haendchen Filho
(Independent IT consultant, Brusque, Brazil
https://orcid.org/0000-0002-7998-8474, aluizio.h.filho@gmail.com)

Adson Marques da Silva Esteves
(Universidade do Vale do Itajaí, Itajaí, Brazil
https://orcid.org/0000-0002-2561-133X, shinadson@gmail.com)

Hércules Antonio do Prado
(Universidade Católica de Brasília, Brasília, Brazil
https://orcid.org/0000-0002-8375-0899, prado.hercules@gmail.com)

Edilson Ferneda
(Universidade Católica de Brasília, Brasília, Brazil
https://orcid.org/0000-0003-4164-5828, eferneda@gmail.com)

André Luis Alice Raabe
(Universidade do Vale do Itajaí, Itajaí, Brazil
https://orcid.org/0000-0002-8093-2507, raabe@univali.br)

Abstract: The high dropout rate in Information Technologies courses is a relevant problem in many countries, mainly because of the increasing demand for professionals in this sector. Usually, high dropout rates in these courses are related to difficulties in algorithms and programming subjects. Content recommendation systems are proposed to mitigate this problem, employing adaptive learning environments that facilitate the learning process. This study presents a content recommendation system that uses learning paths to group students and provide personalized recommendations based on peers' progress. The work follows the many efforts of group-based recommendation systems reported in the literature. The system uses intelligent agents and clustering algorithms to implement the recommendation system and was evaluated by submitting the simulation results to the judgment of human experts who significantly agreed with them. This initiative could make programming teaching more adaptive, using the groups' knowledge. Facilitating learning is one of the key issues to reduce dropout rates and resolve the shortage of labor in the technological area in Portuguese-speaking countries.

Keywords: Recommendation System, Recommender System, Adaptive Learning, Clustering, Portugol Studio
Categories: D.0, H.4, L.2
DOI: 10.3897/jucs.115016

1 Introduction

A growing shortage of Information Technology (IT) professionals poses a significant challenge in Brazil. Over the next five years, approximately 800,000 new jobs are expected to be generated, while only nearly 53,000 professionals are graduating annually, resulting in a staggering job deficit of 532,000 [Kochhann, 21; INEP, 20; MEC-IBGE, 20; G1-JN, 22; Muraro, 23]. On the other hand, around 150,000 new students enroll annually in computer science courses offered by Brazilian universities [MEC-IBGE, 20], pointing to a significant dropout rate. Researchers indicate that one of the main reasons for this evasion is the difficulty found in the initial logic and programming disciplines [Noschang, 14; Kori, 15; Lobo, 17; Hoed, 18]. These difficulties can make advanced disciplines even more challenging and demotivating, ultimately leading to students dropping out.

To address this issue, the Portugol Studio (PS) platform was developed at LITE-UNIVALI [Noschang, 14; Esteves, 19]. With nearly one million downloads, PS is widely used to teach logic and programming in Brazil and Portuguese-speaking countries. Its popularity can be attributed to its user-friendly interface and easy syntax, enabling users to write algorithms and programs simply and intuitively using commands and receiving error messages in Portuguese. The most frequent requests from the PS user community are: *(i)* enabling students to choose their preferred content more flexibly, and *(ii)* providing anywhere access to the platform by the student (the current version must be installed on a desktop computer). The research results presented in this paper address the community's requests.

The PS users' community asks for more flexible content management, which can be provided by including PS IDE, a Learning Management System (LMS) specifically designed for programming beginners. The platform's remote access is also solved by refactoring PS in the web environment. The LMS proposed is named Learning Path Platform (LPP) and includes: *(i)* a content manager that teachers can operate and *(ii)* an adaptive recommendation (or recommender) system that provides personalized recommendations based on the characteristics, preferences, and learning paths [Xu, 23] of the students.

Based on the NFL Theorem [Wolpert, 96; Wolpert, 97], which shows that no learning algorithm is superior, the platform allows different algorithms to search for the best response. In this work, the Latent Dirichlet Allocation (LDA) algorithm [Blei, 02] was chosen to group the students. This algorithm is also commonly adopted by the research community in recommendation systems.

The recommendation system has been designed to operate within an LMS, assisting beginners in their learning paths by offering tailored content recommendations. Unlike previous studies [Raj, 22], focused on providing recommendations based on individual student progress within a specific subject, this article presents an approach that takes advantage of shared learning interests between groups of students. In this sense, instead of relying solely on interests and hobbies, the proposed strategy incorporates learning paths to group students, enabling the system to provide content recommendations that align with the topics already studied by their peers.

The research methodology is hypothetical-deductive, involving the specification of the domain problem, a hypothesis for solving the problem, the creation of an empirical application, a data collection step, and the analysis of results. Simulations demonstrate the effectiveness of using clustering to create groups of students and generate personalized suggestions aligned with shared learning interests. To evaluate the proposed approach, the recommendation algorithm's results were submitted to experts who significantly agreed with them (Figure 6). Results matched predicted recommendations, strengthening the rationale that group preferences enrich recommendations better than those based on standard trajectories.

2 Background

This section provides an overview of the resources involved in the proposed solution, including a brief description of PS and how the LDA algorithm works.

2.1 Portugol Studio

Portugol Studio is a beginner-oriented programming Integrated Development Environment (IDE) with significant recognition in logic and programming teaching [Noschang, 14; Esteves, 19]. Developed at LITE-UNIVALI, this open-source platform has been downloaded more than 800k times and is widely adopted for teaching logic and programming in Brazil and Portuguese-speaking countries. PS's popularity can be attributed to its user-friendly interface and straightforward syntax, allowing users to write algorithms and programs effortlessly. Notably, PS stands out for its unique feature of providing error feedback in Portuguese, thus improving students' experience.

PS was developed to contribute to efforts to combat dropout in logic and programming disciplines in Brazilian universities. It abstracts away necessary complexities found in programming languages, such as syntax, layout, and language, while providing features to simplify the debugging process with more understandable error messages. PS also offers a variety of resources for IDEs, including an interpreter, debugger, examples, libraries, help, tutorials, and support videos.

Statistical data have shown alarming dropout rates in the first stages of computer science courses. Professionals in teaching programming have identified several possible causes for this phenomenon [Noschang, 14]. These causes include *(i)* the use of English syntax and graphical interfaces, which can be a barrier for non-native English speakers; *(ii)* overwhelming graphical interfaces with numerous elements such as menus, settings, and windows, which can intimidate users; *(iii)* error messages that are difficult to understand; *(iv)* the complexity of using and configuring debuggers; and *(v)* the intricate syntax of programming languages [Esteves, 19].

2.2 LDA algorithm

LDA is a three-level hierarchical Bayesian model introduced by [Blei, 02]. It is a generative statistical model used to analyze collections of discrete data, mainly text corpora. LDA represents each item in the group as a mixture of topics along with their probabilities. It is a powerful tool for uncovering hidden thematic structures within a collection of documents [Griffiths, 07]. By estimating the topic probabilities for each document, LDA allows one to envision the underlying themes and relationships among different pieces of text [Wei, 06]. This information can be valuable for many applications, including document clustering, topic modeling [Steyvers, 04], information retrieval, and clustering for recommendation [Apaza, 14; Erkens, 16; Lin, 21].

Efficient approximate inference techniques, such as variational methods and the expectation-maximization (EM) algorithm, are commonly used to estimate the parameters of the Bayesian model. These techniques enable the estimation of the topic distributions and topic-word distributions, which capture the relationships between topics and words in the corpus.

LDA has been applied in many domains, including document modeling, text classification, and collaborative filtering. Researchers have compared LDA with other models to evaluate performance across different scenarios: *(i)* [Papadia, 15] compared LDA, Non-negative Matrix Factorization (NMF), Neural LDA, and Correlated Topic Model for summarizing key topics from large datasets in the context of call center conversations; *(ii)* [Egger, 22] evaluated LDA, NMF, Top2Vec, and BERTopic for topic modeling; and *(iii)* [Chen, 17] compared Latent Semantic Analysis (LSA), Probabilistic LSA, and LDA in the context of semantic modeling for discovering latent document topics.

3 Related works

The related works are analyzed under three points of view: *(i)* the existence of IDE in Portuguese for programming beginners, *(ii)* adaptive recommendation systems in the educational context, and *(iii)* the group-based approaches for recommendation systems.

3.1 Portuguese Programming IDEs for Beginners

The literature presents several environments focused on teaching programming [Cooper, 00; Resnick, 09; Wolber, 11; Paiva, 16; Romagosa, 19]. However, the available options (e.g., [Resnick, 09; Cabada, 18; Romagosa, 19]) are complex for beginners and unintuitive, leading to long learning curves. Furthermore, the option to offer error and debug messages only in English makes it difficult to use for beginners in programming who speak Portuguese.

Next, two approaches are discussed that, like PS, also adopt the Portuguese language: Egua Programming Language (LEP) [Neves, 20] and Portugol WebStudio [Silva, 22].

LEP is an open-source environment for teaching Mathematics and Natural Sciences using computational models. LEP applies a socio-constructivist approach that enables learners to observe algorithm execution and engage with arithmetic operations. LEP's strengths are its ability to boost engagement and comprehension for beginners learning core programming concepts and applying them to math and science. It promotes more interest, deeper understanding, and critical analysis and reflection.

Portugol WebStudio is a platform developed over the source code of PS, invoked from a TypeScript code. Portugol WebStudio does not require installation since its code is executed locally in the user's browser. This characteristic was achieved by refactoring the PS code into JavaScript.

3.2 Adaptive recommendation systems

Adaptive Intelligent Tutorial Systems (AITS) are educational resources that aim to reduce difficulties in specific courses by providing personalized guidance. AITS considers each student's characteristics while supporting their learning paths. These systems create learning environments that store and view relevant student information to offer tailored advice [Lemke, 14; Kataris, 21]. The software is based on student needs and Artificial Intelligence to support learning and teaching. Common strategies include navigating tailored learning paths, adapting learning objects, offering helpful tips, and providing personalized recommendations. The content recommendation algorithm steers students toward optimal learning paths matching their profiles and needs.

Adaptive e-Learning Systems (AES) focus on personalized learning [El-Sabagh, 21; Alqahtani, 21; Alshammari, 19]. An AES aims to individualize the learning process using student-centered instruction tailored to their unique characteristics and preferences. It is an up-and-coming model compared to commonly used reactive e-learning systems [Alqahtani, 21]. AES offers personalized teaching, learning content adapted to different circumstances, and customized recommendations.

The approach presented by [Diao, 22] involves: *(i)* generating concept maps for different types of students considering the degree of mastery of a set of students over the concepts involved; *(ii)* generating a set of weak concept learning paths using a classification procedure; *(iii)* apply an LSTM+attention mechanism to predict the effect of these weak concepts and *(iv)* to select weak concept paths as personalized recommendations. The method's effectiveness is evaluated by comparing its results in an experimental database with recommendations based on grouping, content, and student.

3.3 Recommendation systems based on group knowledge

Group-based recommendation systems are a research field with many applications and approaches. [Uta, 24] argues that knowledge-based group recommendation systems [Pessemier, 17; Felfernig, 24] are typically built on one of the following approaches: *(i)* aggregated predictions or *(ii)* aggregated models. In the case of aggregated predictions, recommendations for individual group members are consolidated, whereas aggregated models merge the preferences of individual group members, forming a cohesive group preference model used for determining recommendations. In the context of aggregated models, as adopted in this work, the preferences of group members are used to build a group preference model, which is then used to determine recommendations.

Group recommendation systems facilitate the collective determination of recommendations for entire groups rather than individual users. These systems employ fundamental recommendation techniques to ensure that recommendations align with the preferences of all, or at least a subset, of group members. For instance, the average user rating per item can be seen as a collective item rating provided by the entire group.

When a recommendation system is built on aggregated models, individual user preferences must be mixed into a group model, which is then employed for recommendation purposes. It is worth noting that employing aggregate models may lead to a situation where it is impossible to obtain an exact match between the group model's attribute settings and the learning items' corresponding properties. However, in such cases, the applied similarity functions will suggest items that closely resemble the preferences stored in the group model.

This work relies on aggregated recommendation models for the educational context, which have motivated many research efforts [Borges, 18; Dwivedi, 15; Srba, 15; Krouska, 20; Ramos, 21; Diao, 22]. [Diao, 22], is of particular interest to this work due to the superior results it obtained compared to its predecessors using groups created with the *k*-means algorithm.

3.4 Application of LDA for creating similarities in groups

LDA has gained considerable attention for its ability to identify similarities and extract meaningful topics from large datasets across various educational contexts. [Apaza, 14] applied LDA to discover the primary issues of courses in an online course recommendation system, specifically for Massive Open Online Courses (MOOCs). The recommendation system can suggest relevant MOOCs by analyzing students' grades and comparing their topic preferences to those discovered by LDA.

The Grouping and Representing Tool (GRT), introduced by [Erkens 16], enables collaborative learning by identifying heterogeneous student groups that allow a sharing process of experiences. GRT combines text mining and LDA to analyze texts written by students throughout their academic journey. By surfacing student similarities and differences, LDA helps form heterogeneous course groups expected to provide impactful collaborative experiences.

Applying AI techniques, [Lin, 21] proposed personalized educational resources based on students' needs, hobbies, and interests. The LDA has been incorporated into a three-tiered Bayesian model and assesses student preferences and requirements by evaluating online course performance (grades and duration).

4 Proposal

This proposal adopts learning objects (LO), a primarily used structure for representing contents in virtual learning systems [Tarouco, 03; Tarouco, 14; Alarcón, 15]. LO refers to educational content designed and organized in small, self-contained sets to enhance learning. It represents comprehensive educational units that can be reused and adapted for many learning purposes. They may have diverse media components, including text, images, audio, and video.

PS is currently under a transition from a desktop to a web-based version. Its IDE has been enhanced by including an LMS, which offers the following features:

- A robust and user-friendly platform for managing and delivering learning materials and resources.
- Enhanced collaboration: the web version of PS facilitates collaboration among users by enabling features such as discussion forums, project sharing, and real-time collaboration on coding tasks.
- Allows the teacher to manage content (e.g., courses and themes) and the students to choose the learning objects that best suit their interests.
- Empowering students to personalize their learning experience suggests learning content based on their preferences, allowing them to select materials and create tailored learning paths.
- Yet, the user can choose a standard structured curriculum instead of an adaptive recommendation in a more flexible learning experience.

The LMS (Figure 1) includes the following models: *(i)* Content Management Model, which handles and organizes learning materials; *(ii)* Recommendation Model, which provides personalized recommendations and guides students through tailored learning paths; and *(iii)* Interaction Model, which facilitates engagement procedures like chats and forums.

The recommendation model is the core component and primary focus of this work. Its main objective is to enhance the learning experience for PS users. However, considering the relationship of the recommendation model to the entire architecture, an overview of the elements of the LMS architecture is needed.

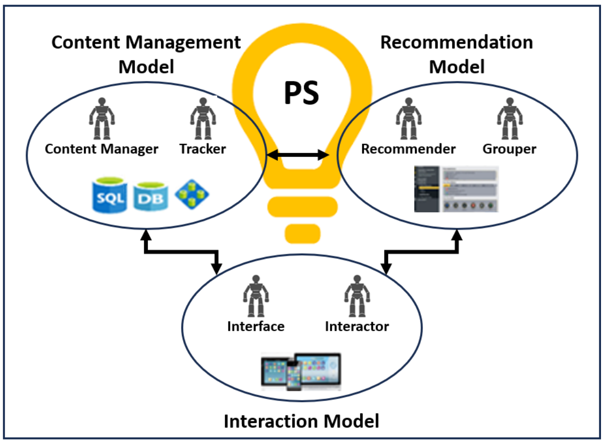

Figure 1: LMS generic architecture

The LMS architecture involves a set of software agents responsible for collaborating and sharing services to provide an adaptive, engaging, and dynamic learning experience. Below, these agents are presented along with some of the tasks for which they are responsible for:

(i) `Content_Manager`: responsible for content management, covering the creation and storage of courses, learning units, themes, and LO. It uses three types of databases: a relational database for entity information, a document-oriented (NoSQL) database for LO, and a graph-oriented database for storing students' learning paths.

(ii) `Tracker`: carries out follow-up procedures for student actions, including the learning paths taken and objects consumed, the time spent on each activity, and pending and completed assessments.

(iii) `Grouper`: responsible for dynamically creating and updating similarity groups, considering the progress in their learning paths, profiles, age, and so on.

(iv) `Recommender`: responsible for recommending LO by considering the student learning level (applying Bloom Taxonomy) and the most frequent peer choices (using similarity groups).

(v) `Interface`: facilitates interaction among the actors, collects student attributes, displays recommended or consumed LO, creates and presents dashboards, facilitates peer interaction in forums and chats, and performs exercises in a specific domain.

The recommendation proposal involves the following methodological procedures: *(i)* Collecting Data, *(ii)* Running the Grouping and Recommendation Algorithms, *(iii)* Results Analysis, and *(iv)* Evaluation.

4.1 Collecting Data

4.1.1 Creating a course and populating a repository with related LO

A hypothetical course of Introduction to Programming, containing five integrated learning units (LU), topics, and a set of LOs, was created to simulate a real-world scenario. Figure 2 illustrates the content structure defined for this simulation. The collection of LOs needs to be included for simplicity, as it is unnecessary to understand the proposal.

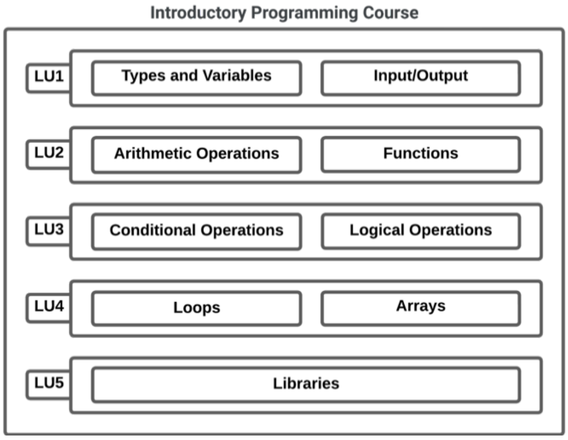

Figure 2: Structure of the Introductory Programming Course

Table 1 presents the LO attributes: *(i)* Name - a self-explanatory field; *(ii)* Description - a summary of what the LO represents or accomplishes; *(iii)* Learning Unit and Topic - are foreign keys that indicate the LO's association with a specific learning unit and topic; *(iv)* Exercise - specifies whether the LO is used in evaluation processes; *(v)* Taxonomy - indicates the level of Bloom taxonomy the LO belongs to; and *(vi)* Tags - themes related to the LO, chosen from the following set (limited to 5 themes):

{*animals*, *animation*, *biology*, *books*, *cars*, *cartoons*, *comedy*, *culture*, *food*, *games*, *geography*, *history*, *image*, *international*, *language*, *marvel*, *math*, *memes*, *monsters*, *movies*, *music*, *myths*, *pets*, *science*, *sports*, *superheroes*, *technology*, *toys*, *YouTube*}

Label	Random string
Description	Random string
Leaning Unit (LU)	Value between 1 and 5
Topic	Value between 15 topics
Exercise	'True' or 'False'
Taxonomy	Value between 1 and 6 (BT levels)
Tags	A value from a list of themes

Table 1: Variables used at dummy LO creation

4.1.2 Creating students and populating a repository

Students must register and provide their profiles, including zip code, date of birth, gender, and general preferences to use the system. This information can be obtained automatically by integrating the LMS with an academic information system. A student repository was populated by randomly generating 200 student profiles with attributes outlined in Table 2. However, students may access the exact LO multiple times; in practice, the simulation considered single activations to simplify modeling. The attribute Tag contains three to five preferences extracted from the set of themes.

Name	Random string
Password	Random string
Gender	Male, Female, …
Age	Value between 12 and 42
Scholar grade	Middle School, High School, College
Tags	A value from a list of themes
Path	A list with the consumed LO

Table 2: Student attributes considered in the simulation.

The first five attributes are self-explanatory. For *Age* attribute, the range of ages was considered from 12 to 42 years for simulation purposes. The three remaining attributes have the following meaning: *(i) Scholar grade* denotes the grade in which the student is currently enrolled; it enables the system to determine the level of logical-mathematical knowledge held by the student and can influence the difficulty level at which problems are presented; *(ii) Tags* comprises the list of different interests the student has selected within the system; it identifies students with similar interests and retrieves content that encompasses topics of interest to the student; and *(iii) Path* encompasses all the LO that the student has used in his learning path, allowing for the identification of his current knowledge; it also enables the system to discern differences in the paths taken by students within the same group, thereby facilitating new recommendations.

4.2 Running the Grouping and Recommendation Algorithms

The student grouping and recommendation algorithms are executed in this step. In practice, this process shall be performed daily to accommodate new students and reassign students who have advanced to higher levels.

The LDA algorithm determined student similarity groups by its proven ability to cluster documents into topics [Blei, 02]. Student attributes were encoded into document-like structures to enable LDA-based grouping, with each feature represented as a string. Specifically, learning paths were transformed from ordered lists into lines by concatenating all LO names, preserving the respective positions in the list. The LDA parameter *k* was set to create groups averaging ten students, with a minimum of two groups, dividing the total student count by ten and rounding up.

Using LDA, these groups generated by the `Grouper` agent were based on specific attributes weighted according to their importance for the similarity calculation. The Path attribute was heavily weighted (5x) if students in a group would likely share OA.

Preferences also received significant weights (3x), while Education and Date of Birth received lower weights (1x). With these groups of students tuned in, personalized recommendations can be generated using the procedures described next.

The initial recommendation for new students is based on your profile and preferences. For regular students, the recommendation considers their profile, preferences, and learning trajectories. This allows students to select the LO that best fits their interests. Furthermore, these LOs are organized based on the student's level in Bloom's Taxonomy (BT) [Bloom, 84; Athanassiou, 03].

The `Recommender` agent manages the recommendation algorithm and works as follows:

For each group:

1. Analyze the learning paths of each student, identifying the LO they have already studied at which levels of BT, and build a list of learning gaps (taxonomy level and LO) for each student.
2. Create a LO ranking of recommendations with the respective frequency.
3. Match student gaps with the most recommended LO and create a list of LO candidates for each student.
4. Classify the LO candidates based on the student's preferences, prioritize those that best suit their interests, and add them to the recommendations list.
5. Determine which LO the student still needs to consume to allow them to complete the lowest levels of BT. This prevents the student from receiving content recommendations for levels higher than the ones he didn't achieve. If no content is available to complete, the next level's content is recommended.
6. Assign ratings to the LO in the recommendation list: *(i)* content with the recommended taxonomy receives a 10-point rating, *(ii)* content with tags that match the student's preferences earns one point each, and *(iii)* the content that other students in the group have on their learning paths earns one point for each student. To allow greater flexibility for the student, the entire set of LOs chosen by their peers but not part of their learning path is included in the list of recommendations with a low rating as information on the possibilities of choice.
7. Sort the list of recommended LO based on the inverse score.

As the student consumes more LO, the importance of the path attribute in the scoring for grouping increases. In real-time, the top 10 contents from this list are then forwarded to the `Interface Agent`, which takes them to display on the student's dashboard.

4.3 Results analysis

The following values were obtained from the simulation for the adopted metrics:

- *Group relevance*: This metric assesses the extent to which the recommended content matches individuals' and groups' preferences and learning paths.
- *Best content*: The best recommendation is based on the highest number of matches with the student's preferences.
- *Recommendations relevance*: this metric represents the overall average relevance of each recommendation. A recommendation is considered relevant when it serves the most significant number of students within each group, considering their

preferences and learning paths. This metric expresses the *calculated* recommendation precision to the *expected* recommendation.

Figures 4, 5, and 6 present the calculated metrics for the number of students interacting with the system across the three setups (100, 200, and 500 LO).

To assess the alignment between a recommendation and the student's interests, as indicated by their preferences and learning paths, a specific LO (Clean DB Search) was taken into consideration. The results of this evaluation are presented in Figure 5.

The analysis begins by examining the student's learning path and identifying the topics they must cover. The LO is then filtered based on these topics and arranged in descending order of the number of tags that match the student's requirements. The system ensures these LOs align closely with the student's BT learning level.

Instances of repeated recommendations within the topics were observed. This phenomenon occurs when a student, despite receiving a recommendation for a particular LO, chooses to select a different LO instead of the recommended one. Consequently, the system recommends the previous content as it remains the most suitable option.

The superior performance of the 200 LO setup can be attributed primarily to the more significant number of preference options available for students to choose from the set of themes. This highlights the need for more LOs, exceeding 100, to better cater to the student's requirements. However, with a group relevance index of 35% (Figure 4), it is ensured that the recommendation will satisfy at least one of the student's preferences.

Figure 6 displays the relevance of the recommendations. The relevance of each recommendation was calculated individually, and the average of these values determined the relevance of the overall recommendations.

The recommendation algorithm considers the objects consumed in the group's learning trajectory, as described in steps two and three of the recommendation algorithm. For new students who still need a learning path, preferences are considered for recommendations. It is important to note that the algorithm generates a list of the ten most consumed LO per group. This list selects LOs based on their alignment with student preferences and learning paths. The resulting value is then converted into a percentage to indicate the relevance of the recommended content to the student.

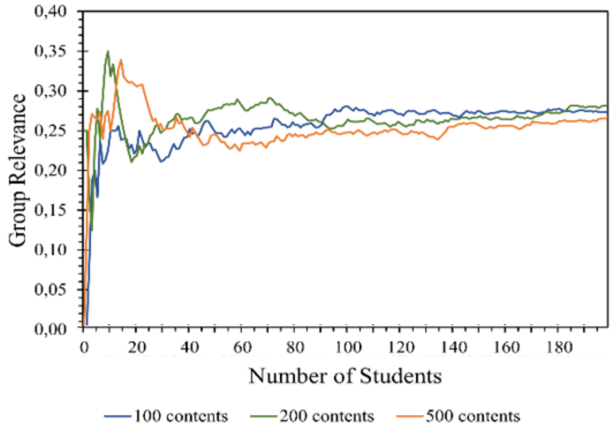

Figure 4: Group relevance

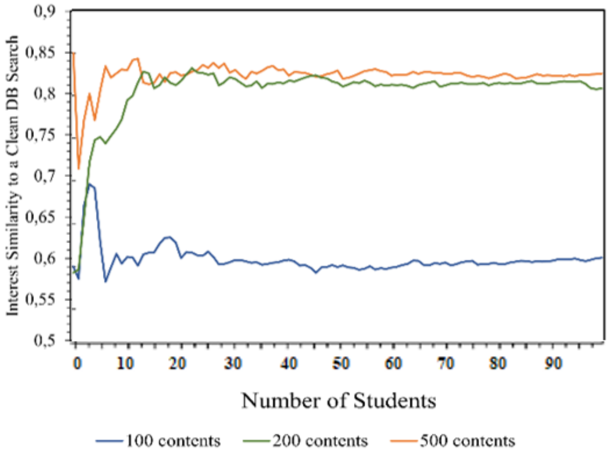

Figure 5: Adherence to the best-recommended content

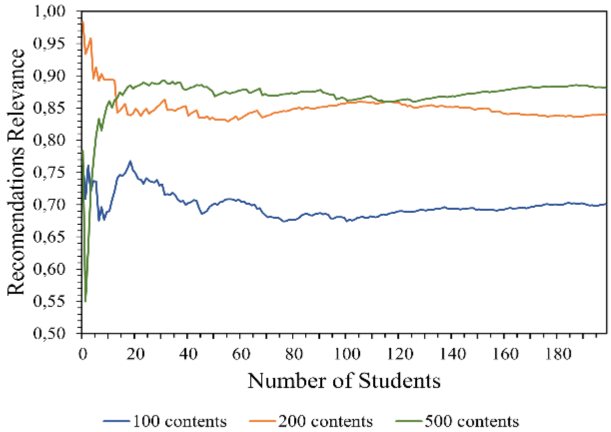

Figure 6: Recommendations relevance

4.4 Evaluation

The group relevance metric is determined by $n/10$, where n represents the average of students who have matched the group's most common tags (maximum of 10 matches). For example, consider a group of three students with 2, 4, and 3 games, respectively. The average is calculated as 3, resulting in a group relevance score of 0.3 (Figure 4). For this simulation, a relevance score ranging from 0.25 to 0.35 is reasonable since each student was allowed to choose only one preference instead of the 3-5 possibilities. In real-world scenarios, it is reasonable to find higher adherence rates. For example, a student who likes Marvel is probably also interested in superheroes, which causes many students to display both preferences on their profiles. Such correlations are not possible with the randomly generated students. The majority of recommendations (88%) issued in the best case (500 LO) were certified by the experts (Figure 6).

5 Discussion and Conclusions

This paper presents an approach for a personalized recommendation system integrated into a virtual learning platform. A simulation validated the algorithm's ability to align recommendations with individual interests based on collective group knowledge. The algorithm's results were submitted to and certified by experts to evaluate the approach.

The system leverages knowledge derived from clusters of students to effectively adapt to dynamic learning needs, surpassing reliance on predetermined patterns. By analyzing the clusters' data, the algorithm can expand the pool of recommended LO as students' knowledge progresses. In other words, the process takes advantage of knowledge from students' learning trajectories in each cluster, allowing for more diverse and enriched suggestions instead of simply following a standard trajectory. Moreover, this work underscores the potential of adaptive, group-based recommendations for improving logic and programming learning and engagement.

Portugol Studio, which has over 856,000 users, was chosen as the target platform. The recommendation approach used agent-based technology to deal with the complexity of this context. Software agents manage content, form groups, and make recommendations, ensuring flexibility and scalability.

The contributions presented in this work are: *(i)* a cluster-based recommendation approach that uses knowledge obtained in learning paths and students' interests to create similarity groups and *(ii)* an adaptive recommendation system integrated into a learning environment to optimize engagement. Although the adoption of an IDE is available in Portuguese to validate the proposal, it is adaptable to other languages in countries that face the same problem of evasion in Information Technology and other related courses.

The simulation is done by generating random content data. The source code to empirically reproduce the experiment is available in the following *GitHub* repository: *https://github.com/AdsonEsteves/SACIP/tree/experiments*

The generation of random data occurs in the `DBConnection` class. Within this class, a static main function generates 500 contents within the topics, types, and tags defined in the *dummydata* function. These generated contents are recorded in the database. The source code for this class can be found at: *https://github.com/AdsonEsteves/SACIP/blob/experiments/src/main/java/sacip/sti/components/DBConnection.java*

From the created contents, it's possible to simulate a recommendation system with students entering and leaving, requesting recommendations according to their preferences. The recommendation is implemented in the `RecommenderAgent` class, specifically in the *getConteudosRecomendados* function: *https://github.com/AdsonEsteves/SACIP/blob/experiments/src/main/java/sacip/sti/agents/RecommenderAgent.java*

The user clustering system is implemented in the `GrouperAgent` class: *https://github.com/AdsonEsteves/SACIP/blob/experiments/src/main/java/sacip/sti/agents/GrouperAgent.java*

References

[Alarcón, 15] Alarcón, F., Alemany, M. M. E., Boza, A., Cuenca, L., Gordo, M.L., Fernández-Diego, M., Ruiz, L.: Learning object: definition and classification. In Proc. 7th Int. Conf. on Education and New Learning Technologies, EDULEARN15, Barcelona, Spain, 4479-4488, 2015.

[Alqahtani, 21] Alqahtani, N. D.: Adaptation in the E-learning systems. Techniun Social Sciences Journal, 23(1), 296–313, 2021. https://techniumscience.com/index.php/socialsciences/article/view/4495

[Alshammari, 19] Alshammari, M. T., Qtaish, A.: Effective adaptive e-learning systems according to learning style and knowledge level. Journal of Information Technology Education: Research, 18, 529-547, 2019. https://doi.org/10.28945/4459

[Apaza] Apaza, R. G., Cervantes, E. V., Quispe, L. C., Luna, J. O.: Online courses recommendation based on LDA. In: CEUR Workshop Proceedings, 1318, 42–48, 2014.

[Athanassiou, 03] Athanassiou, N., McNett, J.M., Harvey, C.: Critical thinking in the management classroom: Bloom's Taxonomy as a learning tool. Journal of Management Education, 27(5), 533-555, 2003.

[Blei, 02] Blei, D. M., Ng, A.Y., Jordan, M.T.: Latent Dirichlet Allocation. Advances in Neural Information Processing Systems, 3, 993–1022, 2002. https://doi.org/10.5555/944919.944937

[Bloom, 84] Bloom, B. S., Krathwohl D. R., Masia B. B.: Taxonomy of educational objectives: the classification of educational goals. Longman, New York, 1984.

[Borges, 18] Borges, S., Mizoguchi, R., Bittencourt, I. I., Isotani, S.: Group formation in CSCL: A review of the state of the art. In: Cristea, A.I., Bittencourt, I.I., Lima, F. (Eds). Higher Education for All. From Challenges to Novel Technology-Enhanced Solutions, vol. 832, Cham, Switzerland: Springer, 2018. https://doi.org/10.1007/978-3-319-97934-2_5

[Cabada, 18] Cabada, R. Z., Estrada, M. L. B., Hernández, F. G., Bustillos, R. O., García, C. A. R.: An effective Web 3.0-based learning environment for a programming language. Telematics and Informatics, 35(3), 611-628, 2018. https:/doi.org/10.1016/j.tele.2017.03.005

[Chen, 17] Chen, L.-C.: An effective LDA-based time topic model to improve blog search performance. Knowledge and Information Systems, 54(1), 243-266, 2017.

[Cooper, 00] Cooper, S., Dann, W., Pausch, R. A.: Alice: a 3-D tool for introductory programming concepts. Journal of Computing Sciences in Colleges, 15(5), 107-116, 2000. https://dl.acm.org/doi/10.5555/364133.364161

[Diao, 22] Diao, X., Zeng, Q., Li, L., Duan, H., Zhao, H., & Song, Z. Personalized Learning Path Recommendation Based on Weak Concept Mining. Mobile Information Systems, 2022. https://doi.org/10.1155/2022/2944268

[Dwivedi, 15] Dwivedi, P., Bharadwaj, K. K.: E-learning recommender system for a group of learners based on the unified learner profile approach. Expert Systems, 32(2), 264–276, 2015. https://doi.org/10.1111/exsy.12061.

[Egger, 22] Egger, F., Yu, K.: A topic modeling comparison between LDA, NMF, Top2Vec, and BERTopic to demystify Twitter posts. arXiv preprint arXiv:2201.03180, 2022.

[El-Sabagh, 21] El-Sabagh, H. A.: Adaptive e-learning environment based on learning styles and its impact on development students' engagement. International Journal of Educational Technology in Higher Education, 18, 2021. https://doi.org/10.1186/s41239-021-00289-4

[Erkens, 16] Erkens, M., Bodemer, D., Hoppe, H. U.: Improving collaborative learning in the classroom: Text mining-based grouping and representing. International Journal of Computer-Supported Collaborative Learning, 11(4), 387–415, 2016. https://doi.org/10.1007/s11412-016-9243-5

[Esteves, 19] Esteves, A., Noschang, L., Raabe, A.: Portugol Studio: Towards an open community for research in programming learning. In: XXVII Workshop sobre Educação em Computação, 513-522, 2019. SBC. (In Portuguese)

[Felfernig, 24] Felfernig, A., Boratto, L., Stettinger, M., Tkalcic, M. Group Recommender Systems: An Introduction. Springer Publishing Company Inc., 2024.

[G1-JN, 22] G1-JN.: There are still vacancies in the technology sector in Brazil due to a need for qualified professionals. G1 National Journal. News, 09/29/2022. Available at: https://g1.globo.com/jornal-nacional/noticia/2022/09/29/sobram-vagas-no-setor-de-tecnologia-no-brasil-por-falta-de-profissionais-qualificados.ghtml

[Griffiths, 07] Griffiths, T. L., Steyvers, M., Tenenbaum, J. B.: Topics in semantic representation. Cognitive Science, 31(2), 171-211, 2007.

[Hoed, 18] Hoed, R. M., Ladeira, M., Leite, L. L.: Influence of algorithmic abstraction and mathematical knowledge on dropout rates from Computing degree courses. Journal of the Brazilian Computer Society, 24, 2018. https://doi.org/10.1186/s13173-018-0074-2

[INEP, 20] Instituto Nacional de Estudos e Pesquisas Educacionais Anisio Teixeira – INEP: Technical Summary of the Census of Higher Education 2020, 2020. (In Portuguese) https://download.inep.gov.br/publicacoes/institucionais/estatisticas_e_indicadores/resumo_tecnico_censo_da_educacao_superior_2020.pdf

[Kataris, 21] Kataris, I., Vidakis, N.: Adaptive e-learning systems through learning styles: A review of the literature. Advances in Mobile Learning Educational Research, 1(2), 124-145, 2021. https://doi.org/10.25082/AMLER.2021.02.007

[Kochhann] Kochhann, L. E.: Higher education does not make up for the lack of IT professionals in Brazil: how can this situation be changed? Educational Challenges Portal, Articles, 5/3/2021. Available at: https://desafiosdaeducacao.com.br/ensino-superior-curso-ti/

[Kori, 15] Kori, K. Pedaste, M., Tõnisson, E., Palts, T., Altin, H., Rantsus, R., Sell, R., Murtazin, K.: First-year dropout in ICT studies. In IEEE Global Engineering Education Conf., EDUCON, 437-445, 2015.

[Krouska, 20] Krouska, A., Virvou, M.: An enhanced genetic algorithm for heterogeneous group formation based on multi-characteristics in social-networking-based learning. IEEE Transactions on Learning Technologies, 13(3), 465–476, 2020.

[Lemke, 14] Lemke, C.: Intelligent Adaptive Learning: An Essential Element of 21st Century Teaching and Learning. Dreambox Learning, 2014. https://www.dreambox.com/resources/white-paper/intelligent-adaptive-learning-an-essential-element-of-21st-century-teaching-and-learning.

[Lin, 21] Lin, Q., He, S., Deng, Y.: Method of personalized educational resource recommendation based on LDA and learner's behavior. International Journal of Electrical Engineering Education, 2021. https://doi.org/10.1177/0020720920983511

[Lobo, 17] Lobo, R.: A evasão no ensino superior brasileiro – novos dados. Estadão, 2017. https://educacao.estadao.com.br/blogs/roberto-lobo/497-2/

[MEC-IBGE, 20] MEC-IBGE: Higher Education Census. Statistics Committee, 2020. Available at: https://ces.ibge.gov.br/base-de-dados/metadados/inep/censo-da-educacao-superior.html

[Muraro, 23] Muraro, L.: Deficit of professionals can hold back the growth of the technology sector—a Gazeta - Articles, 13/03/2023. https://www.agazeta.com.br/artigos/deficit-de-profissionais-pode-reter-o-crescimento-do-setor-de-tecnologia

[Neves, 20] Neves, L. P., Costa, H. A. O., Costa, D. C. L., Meneses, L. A.:. EGUA: a Linguagem de Programação Desenvolvida para o Ensino da Matemática. Série Educar: Matemática, Tecnologia, Engenharia. V. 35. Cap. 12. Belo Horizonte, Brazil: Editora Poisson, 2020. https:/doi.org/10.36229/978-65-86127-44-7

[Noschang, 14] Noschang, L. F., Pelz, F., Jesus, E., Raabe, A.: Portugol Studio: an IDE for programming beginners. In: XXII Workshop sobre Educação em Computação, 535–545, 2014. (In Portuguese).

[Paiva, 16] Paiva, J. C., Leal, J. P., Queirós, R. A.: Enki: A Pedagogical Services Aggregator for Learning Programming Languages. In Proc. of the 47th ACM Conf, on Innovation and Technology in Computer Science Education (ITiCSE'16), 2016. https://doi.org/10.1145/2899415.2899441

[Papadia, 15] Papadia, G., Pacella, M., Perrone, M., Giliberti, V.: A comparison of different topic modeling methods through a real case study of Italian customer care. Expert Systems with Applications, 42(17), 6767–6777, 2015.

[Pessemier, 17] Pessemier, T. D., Dhondt, J., Martens, L.: Hybrid group recommendations for a travel service. Multimedia Tools and Applications 76, 2787–2811, 2017. https:/doi.org/10.1007/s11042-016-3265-x.

[Raj, 22] Raj, N. S., Renumol, V. G.: A systematic literature review on adaptive content recommenders in personalized learning environments from 2015 to 2020. Journal of Computing Education, 9, 113–148, 2022. https://doi.org/10.1007/s40692-021-00199-4

[Ramos, 21] Ramos, M. M. D., Ramos, B, Gadelha, B. F., Oliveira, E. H. T.: An Approach to Group Formation in Collaborative Learning Using Learning Paths in Learning Management Systems, IEEE Transactions on Learning Technologies, 14(5), 555–567, 2021. https:/doi.org/10.1109/TLT.2021.3117916

[Resnick, 09] Resnick, M., Maloney, J., Hernández, A. M., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J. S., Silverman, B., Kafai, Y. B.: Scratch: Programming for all. Communications of the ACM, 52(11), 60-67, 2009. https://doi.org/10.1145/1592761.1592779.

[Romagosa, 19] Romagosa i. Carrasquer, B.: The Snap! Programming System. In: Tatnall, A. (eds) Encyclopedia of Education and Information Technologies. Springer, Cham, 2019. https://doi.org/10.1007/978-3-319-60013-0_28-2

[Silva, 22] Silva, D. G. S., Silva, D. G. S., Soussa, M. R. B.: Portugol WebStudio: IDE Online de Desenvolvimento em Portugol como instrumento de ensino-aprendizagem. Ensino e Tecnologia em Revista, 6(1), 16-30, 2022.

[Srba, 15] Srba, I., Bielikova, M.: Dynamic group formation as an approach to collaborative learning support. IEEE Transactions on Learning Technologies, 8(2), 173–186, 2015.

[Steyvers, 04] Steyvers, M., Griffiths, T. L.: Probabilistic topic models. In: Landauer, T. K., McNamara, D. S., Dennis, S., & Kintsch, W. (Eds.). Handbook of latent semantic analysis, 417-440. Mahwah, NJ: Erlbaum, 2004.

[Tarouco, 03] Tarouco, L. M. R., Fabre, M. J. M., Tamusiunas, F. R.: Reusability of educational objects. RENOTE – Revista Novas Tecnologias para a Educação, 1(1), 2003. (in Portuguese) https://doi.org/10.22456/1679-1916.13628

[Tarouco, 14] Tarouco, L. M. R, Costa, V. M., Ávila, B. G., Bez, M. R., Santos, E. F.: Learning Objects: Theory and Practice. Evangraf, Porto Alegre, Brazil, 2014. (in Portuguese) https://www.lume.ufrgs.br/bitstream/handle/10183/102993/000937201.pdf

[Uta, 24] Uta, M. Felfernig, A., Le, V.-M., Tran, T. N. T., Garber, D., Lubos, S., Burgstaller, T.: Knowledge-Based Recommender Systems: Overview and Research Directions. Frontiers in Big Data, 7, 2024. https://doi.org/10.3389/fdata.2024.1304439

[Wei, 06] Wei, X., Croft, W. B.: LDA-based document models for ad-hoc retrieval. In 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2006. https://doi.org/10.1145/1148170.1148204

[Wolber, 11] Wolber, D.: App inventor and real-world motivation. Proceedings of the 42nd ACM Technical Symposium on Computer Science Education. (SIGCSE'11), 2011. https://doi.org/10.1145/1953163.1953329.

[Wolpert, 96] Wolpert, D. H.: The lack of a priori distinctions between learning algorithms. Neural Computation, 8(7), 1341–1390, 1996.

[Wolpert, 97] Wolpert, D. H., Macready, W. G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1), 67–82, 1997.

[Xu, 23] Xu, X., Hong, W. C. H., Zhang, Y., Jiang, H., Liu, J.: Learning paths design in personal learning environments: The impact on postgraduates' cognitive achievements and satisfaction. In: Innovations in Education and Teaching International, 2023. https://doi.org/10.1080/14703297.2023.2189603