electronics

MDPI

Article

Automated Formative Feedback for Algorithm and Data Structure Self-Assessment

Lourdes Araujo *,†, Fernando Lopez-Ostenero †, Laura Plaza † and Juan Martinez-Romo †

Department of Information Languages and Systems, Universidad Nacional de Educación a Distancia (UNED), 28040 Madrid, Spain; flopez@lsi.uned.es (F.L.-O.); lplaza@lsi.uned.es (L.P.); juaner@lsi.uned.es (J.M.-R.)

* Correspondence: lurdes@lsi.uned.es

† These authors contributed equally to this work.

Abstract: Self-evaluation empowers students to progress independently and adapt their pace according to their unique circumstances. A critical facet of self-assessment and personalized learning lies in furnishing learners with formative feedback. This feedback, dispensed following their responses to self-assessment questions, constitutes a pivotal component of formative assessment systems. We hypothesize that it is possible to generate explanations that are useful as formative feedback using different techniques depending on the type of self-assessment question under consideration. This study focuses on a subject taught in a computer science program at a Spanish distance learning university. Specifically, it delves into advanced data structures and algorithmic frameworks, which serve as overarching principles for addressing complex problems. The generation of these explanatory resources hinges on the specific nature of the question at hand, whether theoretical, practical, related to computational cost, or focused on selecting optimal algorithmic approaches. Our work encompasses a thorough analysis of each question type, coupled with tailored solutions for each scenario. To automate this process as much as possible, we leverage natural language processing techniques, incorporating advanced methods of semantic similarity. The results of the assessment of the feedback generated for a subset of theoretical questions validate the effectiveness of the proposed methods, allowing us to seamlessly integrate this feedback into the self-assessment system. According to a survey, students found the resulting tool highly useful.

Keywords: online learning; formative feedback; natural language processing; semantic similarity; large language models

Academic Editor: Hubert Zarzycki

Received: 29 January 2025
Revised: 24 February 2025
Accepted: 1 March 2025
Published: 5 March 2025

Citation: Araujo, L.; López-Ostenero, F.; Plaza, L.; Martinez-Romo, J. Automated Formative Feedback for Algorithm and Data Structure Self-Assessment. *Electronics* **2025**, *14*, 1034. https://doi.org/10.3390/electronics14051034

1. Introduction

Autonomous learning [1] enables learners to take control of their own learning process. A critical aspect of autonomous learning is self-regulation. Students set their own goals, plan their study strategies, and monitor their progress. Because of this need to track progress during the learning process, technology has played a significant role in facilitating autonomous learning. Online resources, educational apps, and digital platforms provide students with access to a vast repository of information and interactive tools to support their learning journeys.

Incorporating autonomous learning into educational institutions often requires a shift in teaching methods and assessment strategies. Educators become facilitators and mentors, guiding students and providing resources rather than delivering lectures. Self-assessment methods may shift toward more formative and qualitative approaches, focusing on the

demonstration of knowledge and skills rather than standardized tests. Autonomous learning is a powerful framework that encourages learners to become self-directed, motivated, and adaptable individuals, ready to thrive in a rapidly changing world. However, the design of a plan for this form of learning must take into account that different learners require different levels of support, so it is important to maximize the available resources so that all learners can find a way forward.

With these considerations in mind, in this work, we develop and apply an AI-based methodology to automatically generate explanatory comments in self-assessment tests, together with the evaluation of linguistic models and the improvement of an educational tool, which is validated through student satisfaction.

Self-assessment tools [2,3] are of paramount importance in the process of autonomous learning, acting as a guide that directs students in their process of self-regulation and self-monitoring. They not only allow students to objectively measure their progress but also encourage constant reflection on their strengths and areas for improvement, which impact cognitive and affective learning processes [4]. These tools foster autonomy by empowering students to make informed decisions about their study strategies, adjust their goals based on their progress, and become active agents in their own learning. Furthermore, by promoting self-reflection, self-assessment tools cultivate meta-cognitive skills that are essential for deep and lasting learning.

Self-assessment tools can take different forms depending on the context in which they are used and their purpose. Among them are rubrics, checklists, peer assessments, and self-assessment tests and quizzes. The usefulness of self-assessment tests or quizzes [5] has been known for many years. Multiple-choice quizzes offer a quick and structured way to evaluate factual knowledge and comprehension.

Two critical aspects of self-assessment tools are their ability to adapt to the learner's needs and the level of feedback they provide. On the one hand, personalization [6] in self-assessment tools is essential to maintain learner interest and motivation. Both the difficulty of the questions and the learner's history must be taken into account in order to offer questions that improve their knowledge without exceeding their capabilities. On the other hand, feedback not only provides students with insight into their current knowledge level but also facilitates their knowledge enhancement and brings them closer to achieving their goals.

Formative feedback [7–9] is provided throughout the learning process, aiming to guide and inform learners in real time to identify areas for improvement and make necessary adjustments in their learning strategies. Various forms of feedback can be provided during the self-assessment process. The most common one informs learners about the correctness of their answers. While this type of feedback plays a crucial role in verifying accuracy, it lacks the capacity to enrich learners' knowledge or offer deeper insights into potential misconceptions. A more enlightening type of feedback is sometimes called supplementary instructional feedback or formative feedback. Apart from confirming the accuracy of the response, it provides tailored feedback explaining the rationale behind an incorrect answer and the validity of a correct one.

Technological advances have ushered in a new era in educational resources. From interactive online platforms to augmented reality tools and adaptive learning systems, technology has revolutionized the way students access and engage with educational content. Technology has expanded the reach of education, transcending geographic boundaries and making learning accessible to a large portion of the population. The utilization of cutting-edge advancements in natural language processing and artificial intelligence techniques has the potential to drive significant breakthroughs in educational tool development. The

automation of information generation, facilitated by these techniques, enables us to tailor these tools more precisely to the specific context and subject in which they are employed.

These considerations form the foundation of our proposal. We created a tailored self-assessment tool designed for an advanced computer science course focusing on data structures and algorithms. This course is offered in two programs at a distance learning university. The tool was provided to students during the 2023–2024 academic year and has been well received by them. It has the potential to significantly enhance the learning process by offering detailed explanations for both correct and incorrect responses. Specifically, our emphasis is on automating the generation of explanations for theoretical questions. To achieve this, we harnessed cutting-edge natural language processing models and techniques.

The main contributions of this work can be summarized as follows:

- We designed a methodology to automatically generate explanations for the different options in the self-assessment of theoretical questions, with foundations that can be extracted from available documents. This methodology, which was applied to a computer science subject on advanced data structures and algorithms, can be applied to any other subject for which the same type of source materials are available.
- Different language models based on neural networks were evaluated to select the most suitable paragraphs as explanations for why an option is correct or incorrect as an answer to a question.
- We performed a comprehensive evaluation of the explanations generated with each of the models, comparing them and using them to generate a database of explanations.
- The results of this research were used to improve a self-assessment tool in which the explanations were integrated, expanding the feedback provided to students.
- We evaluated student satisfaction with the tool using a questionnaire that included questions related to different aspects of the tool and, in particular, the usefulness of the explanations provided by the tool.

The remainder of this paper is organized as follows. Section 2 presents the related works. Section 3 describes the context in which our proposal was developed, including the subject under consideration and the self-assessment tool ASSET, and provides a description of the methodology. Section 4 describes the evaluation framework, the results of the feedback generated using different models, and the results of the questionnaire on the self-assessment bot, conducted among the students of the course where it was used. Section 5 discusses the results obtained. Finally, Section 6 draws the main conclusions and discusses future work.

2. Related Works

In this section, we outline several studies relevant to various aspects of our proposal. First, we examine proposals that emphasize the importance of self-assessment and feedback in learning. Next, we explore studies on various methods for generating formative feedback, which is the main focus of our proposal. Lastly, we introduce the semantic similarity techniques in natural language processing and artificial intelligence that are employed in this work.

2.1. The Role of Feedback in the Learning Process

Self-assessment in higher education plays a key role in fostering student autonomy. It allows students to critically evaluate their own learning progress and identify their strengths and areas for improvement. This process not only enhances their academic performance but also cultivates a deeper understanding of their learning objectives.

Self-assessment is of particular importance in distance higher education, as it offers a multitude of advantages in this context. Given this form of learning often takes place independently, self-assessment allows students to monitor their progress and needs. In this way, students can adapt their study strategies to their individual needs and preferences while optimizing their learning experience, level of knowledge, and academic results. In addition, in the absence of immediate face-to-face feedback, self-assessment serves as a valuable tool for students to measure their progress and mastery of course material, thereby fostering motivation.

Research on self-assessment in higher education has examined its role in various disciplines, highlighting its impact on learning and performance. The literature can be grouped into three key areas: (1) the effect of self-assessment on learning outcomes, (2) the role of feedback in self-assessment, and (3) systematic reviews and models of feedback in self-assessment.

2.1.1. Effect of Self-Assessment on Learning Outcomes

Several studies have explored how self-assessment influences students' academic performance and self-regulated learning (SRL).

Ifenthaler et al. [10] conducted a study at a European university for a Bachelor's course in Economic and Business Education. Their results indicated that students use self-assessments predominantly before summative assessments. Two distinct clusters based on engagement with self-assessments were identified, and this engagement was positively related to performance in the final exam. Yan [2] conducted a study with 98 students enrolled in a one-year Master's program at a teacher education institute in Hong Kong. He concluded that self-assessment is a fundamental skill for self-regulated learning (SRL) and occurs at each SRL phase with different patterns. Similar conclusions have been drawn in other fields, such as mathematics [11], computer science [12], and engineering [13], to name a few.

While these studies affirm the positive relationship between self-assessment and academic achievement, they also highlight variations in its effectiveness depending on discipline and context.

2.1.2. The Role of Feedback in Self-Assessment

Self-assessment is inherently linked to feedback, which helps students refine their understanding and validate their learning progress. In undertaking self-assessment, students inherently seek validation and insight from external sources to refine their ideas and improve their understanding. The feedback loop generated through self-assessment not only provides valuable information about areas of improvement but also validates areas of strength, reinforcing positive learning behaviors. In addition, effective feedback improves the accuracy of self-assessment by providing learners with specific and practical guidance on how to refine their skills and knowledge. Thus, in the self-assessment framework, feedback emerges as a dynamic component that not only validates self-perceptions but also guides academic progress.

Several studies have addressed the importance of the feedback provided by self-assessment tools [14,15]. These studies suggest that feedback not only enhances self-assessment accuracy but also fosters deeper engagement with learning materials.

2.1.3. Systematic Reviews and Models of Feedback in Self-Assessment

Given the increasing significance of feedback in self-learning environments, several studies have synthesized existing research to develop feedback models and frameworks.

In their first review, Lipnevich and Panadero [16] presented fourteen models of feedback, studying the empirical evidence under each of them. In their later review, the

authors [8] compared and organized different typologies of feedback, considering feedback aspects such as content, function, presentation, and source.

Several studies have highlighted the importance of formative feedback in different contexts. Spady and Karge [17] examined the relationship between formative self-assessment and student performance and self-efficacy in online graduate courses. The qualitative results reflected a positive relationship between formative self-assessment and academic performance, and the authors concluded that formative self-assessment provides online educators with a tool to improve course effectiveness. Gálvez-López [18] examined formative feedback from a multicultural point of view. Since today's environment of globalization has led to the emergence of multicultural classrooms, the author presented a literature review of the current state of knowledge of the role culture plays in the provision of formative feedback, including cultural differences, potential conflicts, and mitigation strategies. Finally, Yan et al. [9] reviewed work related to students' perceptions of self-assessment tools. Although the results were inconclusive, they did show some interesting trends, such as the importance of formative feedback in students' perceptions.

2.2. Methods for Creating Feedback

The methods for automatically providing feedback vary depending on the type of feedback being considered. Based on the purpose of feedback, a possible classification adopted in different works [19,20] may be the following: informative, corrective, suggestive, formative, and motivational. Corrective feedback directly informs the learner about the correctness or incorrectness of their response or course of action. Suggestive feedback includes hints and guiding questions, as well as materials to review. Informative feedback involves messages that provide additional information about the task being evaluated. Motivational feedback aims to encourage the learner to continue working on the problem.

Feedback can be one of several types simultaneously. For example, a multiple-choice question that provides an explanation next to the "True/False" answer given is an example of both corrective and informative feedback. This is the case with the feedback presented in this paper.

A key aspect of the effectiveness of the feedback and the adherence of students to the use of the tools is the clarity of the information provided [21–23]. For feedback to be effective, it must be frequent, timely, sufficient, and appropriately detailed. It should also have a clear relationship with the purpose of the task and the evaluation criteria. It must also be clear, understandable, and focus primarily on learning, not just evaluation. Feedback is most effective when it aids learners in recognizing their mistakes and addressing any misconceptions. Learners can seek out the necessary feedback themselves, but providing support in this process can greatly enhance their persistence in mastering a subject.

Different approaches have been used to generate feedback in online learning environments [24]. These approaches can be grouped into three main categories: (1) response-based comparison, which compares student responses with a predefined correct answer; (2) graphical aids, such as dashboards and visual representations, which help students interpret feedback, and (3) NLP-based techniques, which employ natural language processing (NLP) to analyze responses and generate insights. Concerning NLP-based approaches, Trausan-Matu et al. [25] developed a tool to analyze chat conversations and online forums, identifying topics and semantic similarities. Similarly, Ono et al. [26] applied text mining to provide instant feedback in a foreign language course.

2.2.1. Feedback in Programming and Ontology-Based Approaches

Several studies have focused on feedback in programming education. For example, Keuning et al. [27] examined techniques for providing feedback on coding exercises,

primarily analyzing programming steps or comparing results. Duong et al. [28] used programming code similarity analysis to generate formative feedback in introductory programming courses.

Ontology-based feedback approaches have also been explored [29,30]. Many feedback systems rely on pre-coded responses or expert-provided solutions [20], which require significant manual effort and limit their adaptability. Other models [29] customize feedback based on learner characteristics, leveraging ontology construction. Additionally, Chang et al. [31] demonstrated the use of data mining to extract rules from tutoring sessions and represent them in the Web Ontology Language (OWL).

2.2.2. Modern NLP-Based Feedback Generation

Despite their outstanding performance in many NLP tasks, large language models (LLMs), such as GPT and BERT, have limitations in generative tasks, as they may generate plausible but incorrect responses and sometimes fail to accurately recognize or rectify subtle errors in complex contexts. For this reason, we have resorted to another way of using these models.

We focus on generating feedback explanations for a fundamental area of computer science: algorithms and data structures. Rather than using LLMs to generate explanations directly, we aim to leverage them for the precise selection of paragraphs from a reference text that can serve as explanations. This approach mitigates the risk of producing incorrect explanations by ensuring that the selected content is accurate and relevant. Araujo et al. [32] presented a preliminary study of the potential of semantic similarity for the selection of texts related to self-assessment questions.

Our proposal differs from those mentioned above in several aspects, including the area of application, i.e., theoretical questions related to algorithms and data structures. Although other proposals have also focused on computer science, most of them provided feedback on coding exercises. It also differs in the methodology used, as it does not use classical methods such as ontologies but rather LLMs. Yet, unlike other systems that use LLMs to directly obtain the feedback, we use them to make a precise search for the paragraphs of a reference text that serve as an appropriate explanation of the question at hand.

2.3. Semantic Similarity and Language Models

Natural language processing (NLP) has become a cornerstone technology in a world where electronically formatted information extends into all domains, including translations, medical reports, news, and web pages. In recent years, NLP techniques have undergone a profound transformation, largely due to the widespread adoption of deep neural networks [33–35]. These advances have led to unprecedented levels of performance in many applications. The introduction of these models allows problems to be modeled end to end, eliminating the need for feature engineering, which was essential in previous machine learning models, each tailored to a specific task. This change has significantly improved efficiency and performance, marking a major milestone in the field of NLP and opening up new possibilities for the automation and improvement of various processes in linguistic data handling.

A very useful tool in language processing is the analysis of semantic similarity [36]. Semantic similarity is the degree to which two texts convey the same information. Measuring semantic similarity is a challenging task that has many applications, such as information retrieval, text summarization, question answering, text analytics, sentiment analysis, and more.

Approaches for Semantic Similarity Computation

The computation of semantic similarity has been explored through multiple methodologies, which can be broadly categorized into ontology-based approaches, knowledge graph-based methods, corpus-based methods, and neural network-based models.

Ontology-Based Methods: Ontologies, such as WordNet [37], provide structured representations of concepts and their relationships within a domain. These structured resources enable the calculation of semantic similarity by leveraging various metrics, such as path length, information content, and feature overlap [38]. While ontology-based methods provide a solid foundation for similarity computation, they often rely on manually curated structures, limiting their scalability and adaptability to broader contexts.

Knowledge Graph-Based Methods: Knowledge graphs, such as DBpedia [39], extend the idea of ontologies by structuring entities and their relationships into large-scale networks. These methods employ techniques such as graph traversal, graph embedding, and graph neural networks to assess entity similarity [40]. Compared to ontology-based methods, knowledge graphs offer a more dynamic and scalable approach, as they can integrate information from multiple sources and continuously evolve. However, they still rely on predefined entity relationships, which may limit their adaptability to highly nuanced linguistic contexts.

Corpus-Based Methods: Corpus-based techniques determine semantic similarity using large collections of texts from specific domains. Words and phrases are represented as high-dimensional vectors, and similarity is measured using methods such as vector space models, latent semantic analysis (LSA), and topic modeling [41]. While corpus-based methods capture statistical patterns in language use, their effectiveness is constrained by the size and diversity of the training corpus, making it challenging to generalize across different contexts.

Neural Network-Based Methods: The advent of word embeddings has revolutionized semantic similarity computation. Early models such as Word2Vec and GloVe learned word representations from contextual co-occurrence, significantly improving performance in capturing word meaning. More recently, transformer-based models, such as BERT [42], RoBERTa [43], and GPT [44], have further advanced the field by generating contextualized word representations that dynamically adjust based on the surrounding text. These models have surpassed previous approaches in their ability to capture subtle nuances of meaning and contextual dependencies, making them particularly effective for complex NLP tasks, including semantic similarity [45].

Comparative Analysis and Justification for Our Approach

In this paper, we employ semantic similarity methods based on language models such as BERT (Bidirectional Encoder Representations from Transformers) and BETO (Spanish BERT). These methods capture the context and semantics of words and use an encoder–decoder architecture to generate vector representations. Semantic similarity is measured using the cosine similarity between word vectors.

The LLMs we use have been trained with large amounts of text data and can effectively identify relationships between words, phrases, and sentences, even when they are expressed in different ways. This deep understanding enables LLMs to accurately measure semantic similarity, making them invaluable for applications such as the one we present in this work.

In our study, we select BETO, BERT, and RoBERTa due to their strong performance in Spanish-language NLP tasks and their well-documented robustness in handling linguistic nuances specific to Spanish. BETO, a Spanish-specific BERT model, is particularly relevant given its pre-training on large-scale Spanish corpora, which ensures a better understanding of syntactic and semantic structures compared to multilingual models. Similarly, we include

BERT and RoBERTa to leverage their strong contextual representation capabilities while ensuring adaptability to domain-specific fine-tuning.

We deliberately opt against using GPT-based embeddings due to the inherent risks associated with generative models, particularly hallucinations, which are undesirable in our task. Given the sensitivity of misinformation detection, we prioritize models that focus strictly on representation learning rather than generation, thereby minimizing the potential introduction of spurious or misleading information.

Although we acknowledge the advancements in transformer-based architectures, our selection is guided by the need for reliability, interpretability, and performance in Spanish-language contexts. Future work may explore additional models, but ensuring factual accuracy and avoiding unintended distortions remains our primary concern.

3. Materials and Methods

To contextualize the study of automatic feedback generation presented in this paper, in this section, we first describe the subject for which automatic feedback generation is designed, as well as the context in which it is taught. We also present the original tool into which the automatically generated feedback is incorporated.

3.1. The Subject

We focus our research on a computer science topic: algorithms and advanced data structures. This subject is central to the curriculum of computer science degrees.

In computer science, structures represent how data are stored and organized to execute operations efficiently. The data, considered as elements within a specific data structure, can range from simple entities, like integers or strings, to more intricate structures. Common data structures include representations of familiar organizational formats, such as lists, queues, or stacks. However, these structures are covered in a preceding course that also explores the analysis of algorithmic costs. Within the subject under consideration, the topics explored include various data structures, such as hash tables, which efficiently associate keys with values for searches; graphs, which facilitate the representation of data and their connections; and heaps, which are utilized to represent priority queues, where higher-priority elements are extracted before lower-priority ones.

The subject also introduces several algorithmic schemes, such as the greedy approach, which is suitable for problems with a criterion enabling direct solution construction without revisiting previous decisions. However, this method is not universally applicable. Another scheme covered is divide and conquer, which involves the division of complex problems into simpler ones, and the backtracking scheme, which explores all potential solutions until it is established that they cannot be valid. Additional algorithmic schemes included in the subject are dynamic programming, which is tailored to problems allowing the reuse of previously calculated solutions, and the branch and bound scheme, which prunes potential solutions once it is proven that they cannot surpass an existing alternative. Each topic is taught by presenting the general case and illustrating it with an algorithm for a specific problem. Subsequently, other classic problems applying the structure or scheme are showcased for a comprehensive understanding.

Figure 1 shows an illustration of the above-described contents of the computer science subject under consideration. We can see that they are organized around two fundamental parts: (1) the teaching of programming and (2) algorithmic schemes and data structures.

The subject is taught in a large Spanish distance learning university. Specifically, it is taught in the first semester of the second year of two computer science degrees. Since it is a distance learning university, it has a large number of students, around 300. It also has students of different nationalities and ages, since distance learning allows students to

combine studies with work. Specific details about the demographics of the students who used and evaluated the tool presented here are included in Section 4.

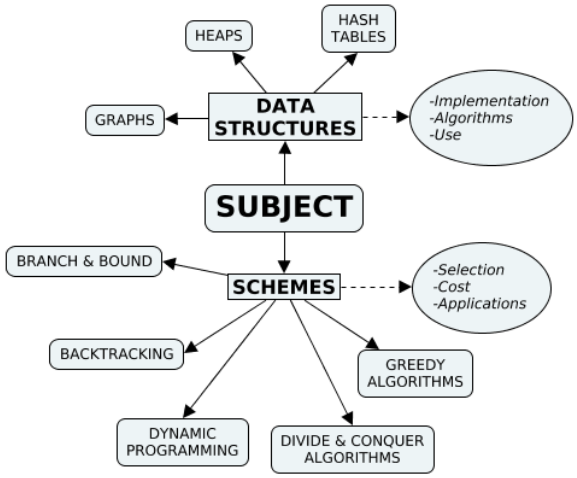


Figure 1. Scheme of the course's contents on algorithms and advanced data structures.

3.2. The Self-Assessment Tool

In previous work, we created a self-assessment tool called ASSET (self-Assessment Students' System with Explanations for their Training), which has been very popular among students. This tool integrates self-assessment functions (multiple-choice questions) with topic-focused navigation, allowing the simultaneous display of the core subject topics and their interconnections, along with self-assessment options for each of these topics. Since its creation, ASSET has been improved by incorporating mechanisms aimed at personalizing and adjusting the challenges presented to the student [46–48]. The next step was the generation of explanations about the answers chosen by the student so that ASSET would become more of a formative tool, not just an assessment tool.

The ASSET tool, which was originally designed as a web-based tool, has recently moved to the Telegram instant messaging platform. This change has led to enhanced ease of access and has enabled significant improvements in the personalization of the tool. Figure 2 shows the bot in Telegram.

A study session consists of an interactive exchange between the user and the bot. The user asks for exercises on a topic and, in response, the bot presents a selection menu, as shown in Figure 2. The user can navigate through the hierarchy of concepts related to the subject and choose a concept to practice.

By choosing a learning concept, such as the data structure "graphs", the tool sends exercises, which can include images and text, to the user (see Figure 3). It also presents several potential answers. The user has the option to skip the exercise and try another one or select one of the possible answers. In the latter case, the tool provides information on whether the chosen answer is correct and offers the option of asking for feedback (see Figure 4).

The personalization mechanisms [48] involve two main aspects. First, the tool takes into account the learner's historical usage to avoid presenting them with questions they have already answered correctly and to focus on areas where they need reinforcement. Figure 5 shows a tool menu offering options to retry failed or skipped exercises, request the next exercise, or obtain statistics on the learner's performance in relation to exercises related to a particular concept. In addition, the user has the option of resetting the exercise history for each concept.

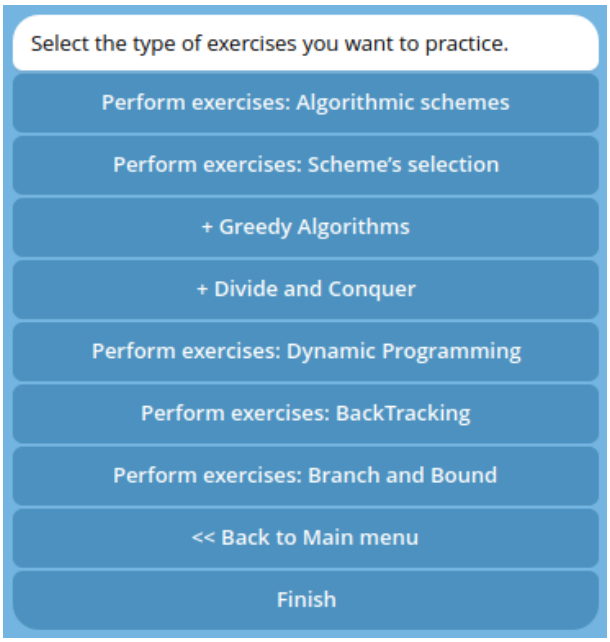


Figure 2. Main menu in the Telegram bot.

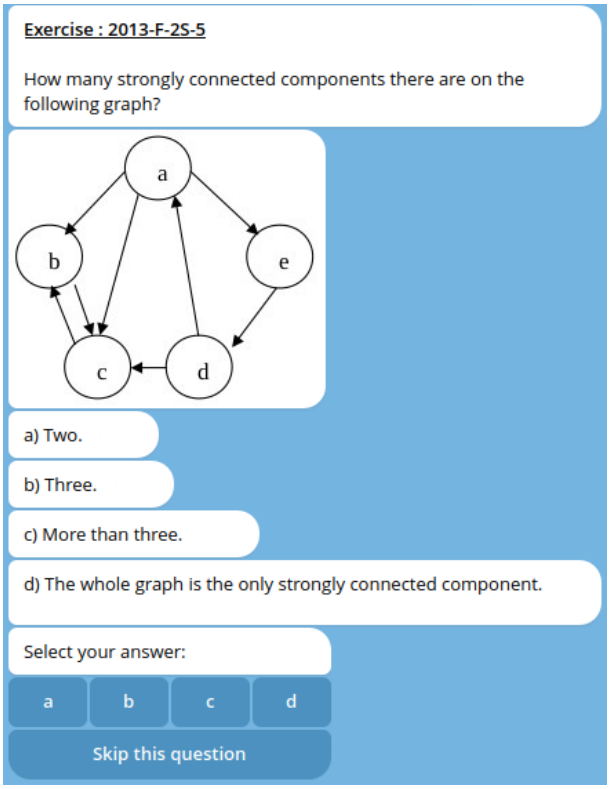


Figure 3. Example of a question in the Telegram bot.

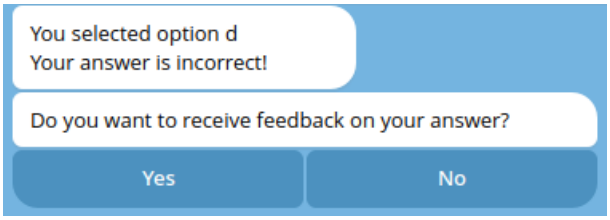


Figure 4. Asking the bot for feedback.

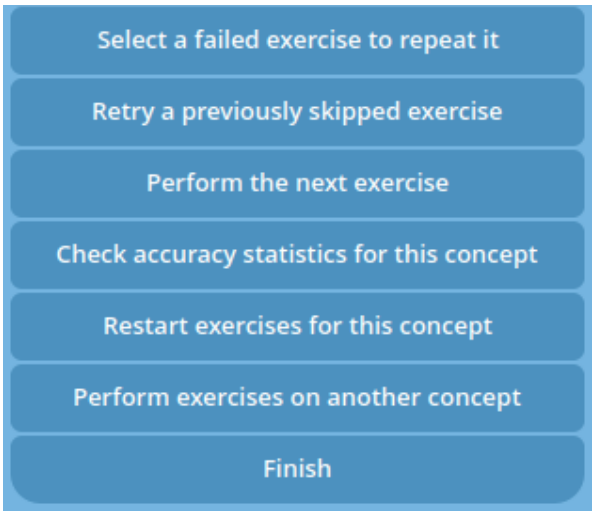


Figure 5. Bot options for deciding what action to take on a received question.

Second, we developed mechanisms to objectively evaluate the difficulty of the questions based on the results of the exams in which they were originally set. The percentage of students who answered each question correctly in the exams provides information on the difficulty of the question, allowing the tool to present the questions to the user in increasing order of difficulty.

As the tool maintains a personalized memory of interactions with each student, it enables study sessions to resume from where they last stopped. This feature provides students with the flexibility to study at their own pace.

Figure 6 shows the information flow of our proposal. The left part of the figure shows the process of generating explanations. Starting from the question under consideration and a set of texts on the same topic as the question, LLMs are used to generate a concise representation of the information in the form of embeddings. Semantic similarity techniques are then used to select the most appropriate explanations for the question. The right part of the figure represents the elements involved in the use of the self-assessment bot. When a question is provided to the user, they select one of the possible answer options. The bot then provides two forms of feedback: an indication of whether the answer is correct and an explanation of the reasons for the correctness or incorrectness. In the following sections, we explain the mechanisms we designed to generate the collection of explanations associated with each question and possible answer.

3.3. Methodology for Feedback Generation

We now present the methodology used for generating feedback.

Self-assessment questions, along with associated information, are stored in XML format. This information includes response options, the correctness of each option, feedback, and a set of labels characterizing the question type. Table 1 shows the set of considered labels and their interpretations. These labels include aspects related to the question type—theoretical, practical, or associated with computational cost—and the topic that is the focus of the question, such as data structures or algorithms. Each question can be assigned multiple labels.

These labels provide the user with the selected question type. Furthermore, they enable us to design feedback generation tailored to each question type. Therefore, we begin by categorizing question types based on the algorithm used for generating feedback. After explaining the most appropriate way to generate feedback for each question type, we focus on automatically generating feedback for different types of theoretical questions using natural language processing technologies.

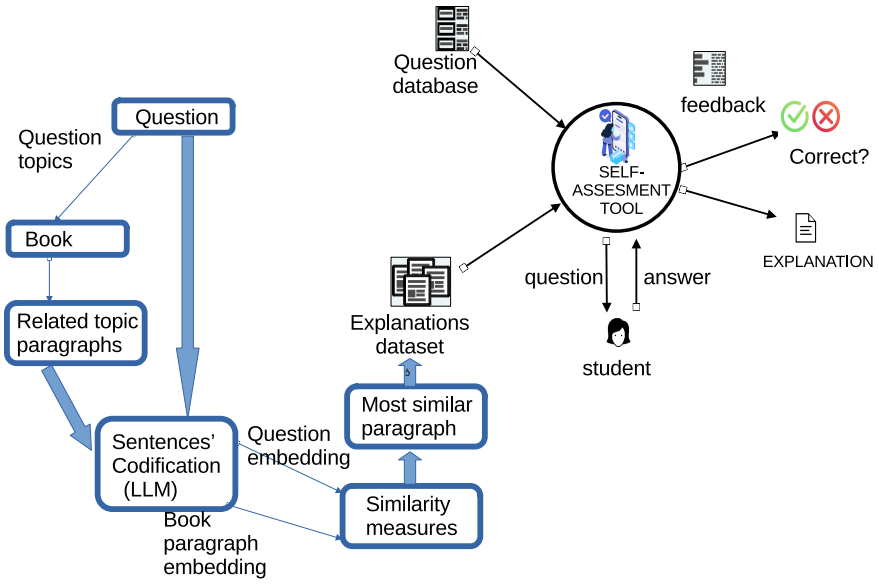


Figure 6. General workflow of the proposal.

Table 1. Hierarchy of labels associated with the main topics and aspects of the field we use to classify them.

PRACTICAL: Practical matter
THEORETICAL: Question of a theoretical nature
COST: Algorithmic cost
DS: Question related to data structures in general
HEAPS: Heaps
GRAPHS: Graphs
HASH: Hash tables
SCHEME: Question related to algorithmic schemes in general
SCH_SEL: Algorithmic schema selection
GREEDY: Greedy scheme
TASK_SCHE: Greedy algorithm for scheduling
PRIM: Prim algorithm
KRUSKAL: Kruskal algorithm
DIJKSTRA: Dijkstra algorithm
D&C: Divide and conquer scheme
QUICKSORT: Quicksort algorithm
DP: Dynamic programming scheme
BA: Backtracking scheme
B&B: Branch and Bound scheme

3.3.1. Question Types

The initial distinction in question types lies in whether they are theoretical or practical. Theoretical questions can be explained based on recommended subject texts and are the primary focus of this study. Practical questions, on the other hand, involve the application of algorithms or data structures to specific data.

The proposal in this paper is not directly applicable to practical issues requiring code execution since it is based on the retrieval of information related to the question under consideration. Therefore, we use another method to generate feedback for practical questions that focuses on presenting the algorithm's trace applied to the problem data.

Within theoretical questions, which are the focus of this work, two aspects are further distinguished: the formulation of the question and the topics within the subject they

address. Regarding formulation, we observed that most questions fall into one of the following categories:

- SPECIFIC SUBJECT: Questions that present different claims about a specific topic. For example, the following question states several possibilities related to the branch-and-bound scheme:
  During the execution of a branch-and-bound scheme, a solution is found that is better than the best existing solution at that time and that improves the optimistic estimate of the top of the heap. This implies that:
  (a) The algorithm has definitely found the solution. (T)
  (b) The bound is updated and the exploration is continued because we are not finished. (F)
  (c) The top of the heap is updated and the exploration is continued because we are not finished. (F)
  (d) None of the other options is correct. (F)
- SCHEME SELECTION: Questions regarding the most suitable algorithmic approach to minimize temporal or spatial costs to solve a problem. For instance, the greedy algorithm is highly effective in addressing problems where the goal is to optimize a specific parameter. Nevertheless, its application is not always feasible. Addressing such problems requires a thorough analysis of how each algorithmic approach could be implemented, whether it would be appropriate, and what the associated costs might be. An illustration of such a query is as follows: *There are n cubes numbered from 1 to n. Each cube has a different letter printed on each of its faces, although different cubes may have repeated letters. Given a word of length n, we want to place the total number of cubes consecutively to form the given word. Which of the following algorithmic schemes is the most appropriate to solve this problem?*
  (a) Dynamic programming. (F)
  (b) Divide and conquer. (F)
  (c) Branch and bound. (F)
  (d) Backtracking. (T)
- COST: Questions related to the algorithmic costs associated with various data structures or algorithms. An example of such a question is as follows: *Among the following, what would be the minimum cost of an algorithm that, given a vector $C[1..n]$ of distinct unordered integers and an integer S, determines whether there exist two elements of C such that their sum is exactly S:*
  (a) O(n). (F)
  (b) $O(n^2)$. (F)
  (c) O(n log n). (T)
  (d) $O(n^2 \log n)$. (F)

3.3.2. Feedback Generation

Based on the question type and whether the option chosen by the user is correct or incorrect, a query is generated. To generate feedback, semantic similarity methods are applied, allowing the identification of paragraphs from reference documents that are most closely related to the query, i.e., those most similar. Specifically, the following cases were identified:

- SPECIFIC SUBJECT:
  - *The chosen option is true*: The query consists of the question text and the option text.
  - *The chosen option is false*: The query consists of the text of the question only. Since the option is false, adding it to the query would introduce noise, lowering the performance of the system.

In both cases, the most relevant paragraphs are retrieved from the documentation for the subject corresponding to the specific topic of the question.

- SCHEME SELECTION:
  - *The chosen option is true*: The query consists of the question text and the option text.
  - *The chosen option is false*: A generic query is composed of terms aimed at selecting explanations of the type of problems for which the scheme is most appropriate, such as "scheme XX applies to problems", "the type of problems for which the scheme is most appropriate", "this schema is appropriate for", "is valid for problems", etc. The purpose of this explanation is to clarify why the selection of this particular scheme was not the correct one.

  In both cases, the most relevant paragraphs are retrieved from the documentation for the topic mentioned in the option text.
- COST: In this case, it is necessary to distinguish whether the question is about the cost of an algorithm or operation using a particular schema or data structures, or whether it is an open-ended cost issue that mentions different schemes or structures.
  - *The question refers to a specific schema or data structure*: The query is formed with the main text of the question and the corresponding option, and the selected answer is classified as correct or incorrect. In addition, specific cost-related terms are added to help in the selection of the most relevant paragraphs: "cost", "O(", "order of", "search space", and "bounded". The objective is to select a paragraph from the reference texts in which the cost of the algorithm under consideration is explained.
  - *The question does not specify the subject matter or includes several subjects*: The query is constructed as in the previous case, but now, the most relevant paragraphs are retrieved from the documentation for all the topics involved, also adding the specific cost-related terms.

In order to compute the similarity between the queries constructed and the reference texts associated with the corresponding topic or topics in each case, we use language models that allow us to represent the texts to be compared and to calculate the similarity between them. Figure 7 shows a scheme of this process.

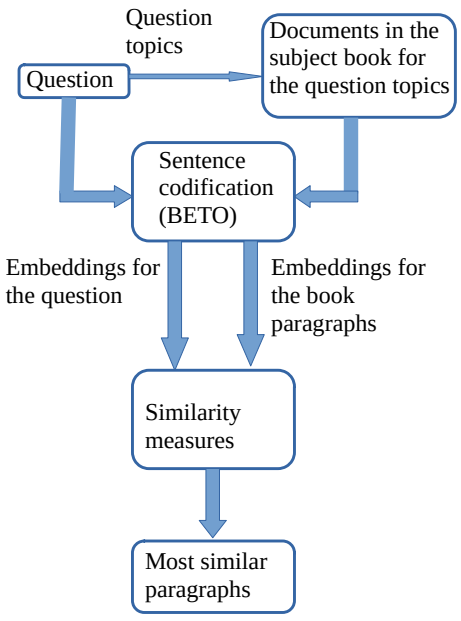


Figure 7. Scheme of the process used to select the most relevant paragraph for feedback.

From the question and each of its possible answers, the topics involved are identified. With these topics, the parts of the reference text related to the question are selected. Both the query and the documents that can provide feedback are transformed into a compact numerical representation or embedding. For this, we use a pre-trained language model trained on large amounts of text. Specifically, we use the BERT model in Spanish (BETO) [49], adapted to the specific task of sentence similarity. From these vector representations, semantic similarity calculations can be applied. In this way, the paragraphs most similar to the question are selected as feedback.

Specifically, the process involves the following steps:

- Tokenization: Both texts are tokenized into smaller units, such as words or subwords. This step is essential to convert the texts into a format that the model can understand.
- Embedding: Each tokenized word or subword is then embedded into a high-dimensional vector space using a pre-trained BERT model. BERT captures contextual information, which implies that the embeddings represent the meaning of words based on their context in the sentence.
- Pooling: The embeddings for each text are aggregated or pooled to generate a fixed-size representation for the entire text. Common pooling methods include mean pooling or max pooling.
- Cosine Similarity: The cosine similarity is then calculated between the two text representations. Cosine similarity measures the cosine of the angle between two vectors, providing a measure of similarity irrespective of the magnitude of the vectors.

We used the Hugging Face Transformers library [50] to perform the previous steps. The similarity metric used is represented by Equation (1):

$$Similarity(a, b) = 1 - cos(\theta) = 1 - \frac{a \cdot b}{||a||||b||} \quad (1)$$

We tested other similarity metrics, such as the Euclidean and Manhattan distances, but they did not improve the results of the cosine similarity.

We tested three freely available BETO-based pre-trained language models, adapted to semantic similarity-related tasks:

- *hiiamsid/sentence_similarity_spanish_es*: This is a sentence-transformer model that maps sentences and paragraphs to a dense 768-dimensional vector space. This model can be used for tasks such as clustering or semantic similarity. Its main advantage is that it has been specially trained for the Spanish language. This model takes as a basis the BETO model (dccuchile/bert-base-spanish-wwm-cased) [51], a BERT model trained on a large corpus in Spanish.
- *hackathon-pln-es/paraphrase-spanish-distilroberta*: This is also a sentence-transformer model that maps sentences and paragraphs to a dense 768-dimensional vector space. The model was developed during the Hackathon 2022 NLP—Spanish, organized by the hackathon-pln-es Organization (https://huggingface.co/hackathon-pln-es accessed on 24 February 2025). It follows a teacher–student transfer learning approach to train a bertin-roberta-base-Spanish model using parallel EN-ES sentence pairs. The strongest available pre-trained English Bi-Encoder (paraphrase-mpnet-base-v2) was used as the teacher model, and the pre-trained Spanish BERTIN was used as the student model.
- *jfarray/Model_dccuchile_bert-base-spanish-wwm-uncased_50_Epochs*: This sentence-transformer model maps sentences and paragraphs to a 256-dimensional dense vector space and can be used for tasks like clustering or semantic search. It is a Spanish BERT model trained on a large Spanish corpus. Its size is similar to that of BERT-Base and was trained using the Whole Word Masking technique.

Each of the selected language models was employed to search for paragraphs to be presented as feedback based on their similarity to the query. The chosen paragraphs were then arranged in descending order of similarity, and for each model, the top five paragraphs most similar to the constructed query were examined.

For the five paragraphs provided by each linguistic model, an evaluation was performed to determine their relevance and potential as feedback. The selected paragraphs were then examined manually. A decision is made on which of them are relevant (RELEVANT), i.e., those that could serve as an explanation for the answer to the question under consideration. Among the relevant ones from the three models, if any existed, the most appropriate one was chosen to be used as an explanation for the corresponding question (SELECTED).

4. Results

In this section, we evaluate the results of our proposal from two different angles. First, we quantitatively evaluate the ability of the selected models to generate adequate explanations as feedback by comparing them using different metrics. Then, we present data on the use of the tool, as well as the results of a student questionnaire about the tool and the explanations it provides.

First, we describe the evaluation metrics used to analyze the results of the system and compare the different models.

4.1. Evaluation Metrics

For evaluation purposes, data were collected for each question and model, such as the number of relevant paragraphs. If the ultimately selected feedback paragraph was among the five, its position in the top list was registered. The position of the first paragraph that was relevant for feedback was also recorded. These data allowed us to compute various metrics regarding the quality of the method.

The metrics used for the system evaluation are some of those most commonly employed to assess information retrieval systems. Actually, the selection of candidate paragraphs for use as feedback was the result of an information retrieval process based on semantic similarity. The metrics used were the following:

- Precision at K (prec@K): Prec@K is used to evaluate the relevance of the top-K items retrieved by a system. It measures the proportion of relevant items among the top-K items. It is computed as shown in Equation (2):

$$P@K = \frac{Number\ of\ relevant\ items\ in\ top\ K}{K} \tag{2}$$

- Hit rate (HR): The HR is the fraction of queries for which the correct answer is included in the recommendation list of length K. The hit rate is 1.0 if there is at least one relevant document among all the top-K retrieved documents. If there are no relevant documents, the hit rate is 0. In our case, this measure is important because it indicates each model's ability to provide at least one paragraph as feedback.
  We calculated two alternative values for the HR in each model. The first, HR_SEL (selected hit rate), was more restrictive and counted as a success when the paragraph ultimately selected for feedback was among those retrieved by the model. The second, HR_REL (relevant hit rate), considered it a success if any relevant paragraph was retrieved for potential feedback, even if it was not ultimately selected.

- Mean reciprocal rank (MRR): The MRR is the multiplicative inverse of the rank of the first correct answer. It is computed as shown in (3) as the average of the reciprocal ranks of results for a sample of queries Q:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{3}$$

  where $|Q|$ denotes the total number of queries and $rank_i$ denotes the rank of the first valid result. This measure is also important as it provides an approximation of the quality of the ranking of responses provided by each model. In our case, we considered the paragraph selected to be used as an explanation as valid.

4.2. Evaluating Feedback Generation

Next, we present the aggregated results for these metrics. Table 2 displays the results obtained for precision at 5 (*Prec*@5). In this table, the first row corresponds to the number of instances of each topic group. The first column shows the overall results for all questions. The following columns show prec@5 for questions corresponding to broad topics for which there are enough cases to have significant results: algorithmic schemes (*Scheme*), the selection of the most appropriate scheme (*Sch_sel*), data structures (*DS*), and algorithmic cost (*Cost*).

Figure 8 shows the corresponding graphical representation. We can observe that the values obtained were quite high for an information retrieval task, with the overall result above 25% for the *hackathon-pln-es* model. Prec@5 values of around 25% indicate that approximately one-quarter of retrieved paragraphs were relevant. In other words, on average, one or two useful paragraphs were retrieved for each query. We need to consider that the values obtained for information retrieval (IR) tasks were typically much lower than those in classification tasks. This is due to inherent differences in the tasks. IR tasks involve retrieving all relevant documents from a potentially vast collection, making it challenging to achieve complete recall. The emphasis is on minimizing false positives. In contrast, classification tasks focus on accurate class assignments among a usually small number of classes.

When comparing the considered models, we can see that the *hackathon-pln-es* model outperformed the other two. These results indicate that the teacher–student transfer learning approach followed by this model may be advantageous over other techniques. The *hiiamsid* model, which was the second-best-performing model, also uses a dense 768-dimensional vector. In last place was the *jfarray* model, which, unlike the previous ones, uses shorter vectors of length 256. However, even for the latter, the values indicate that it often retrieved useful paragraphs.

Regarding the considered topics, we can see that the results for questions about selecting the most appropriate scheme to solve a problem stood out. We believe that this is due to the fact that there are different alternative texts that can justify the application of a certain schema to a problem, thus increasing the probability of retrieving one of them. The worst results were obtained for data structures and computational cost. In the case of data structures, the situation was the opposite of that for schema selection. In many cases, we are dealing with specific definitions or detailed features of the structure that have only one possible explanation. In the case of computational cost, besides the fact that there are considerably fewer instances, it was not possible to find explanations of erroneous cost propositions in the reference texts. In any case, this is not the most appropriate metric for evaluating our system, since for many questions, the five relevant paragraphs to be retrieved do not exist in the reference texts.

Table 2. Precision@5 for the considered models. The first column corresponds to the overall value. The other columns show the results for some of the highest-level topics for which there are more than 35 questions. The first row shows the number of instances in each group. Sch_sel stands for scheme selection and DS for data structures.

Model	Overall	Topic			
		Scheme	Sch_sel	DS	Cost
Instances	182	133	53	58	36
hackathon-pln-es	0.270	0.249	0.415	0.181	0.111
hiiamsid	0.219	0.235	0.577	0.155	0.135
jfarray	0.176	0.150	0.279	0.090	0.081

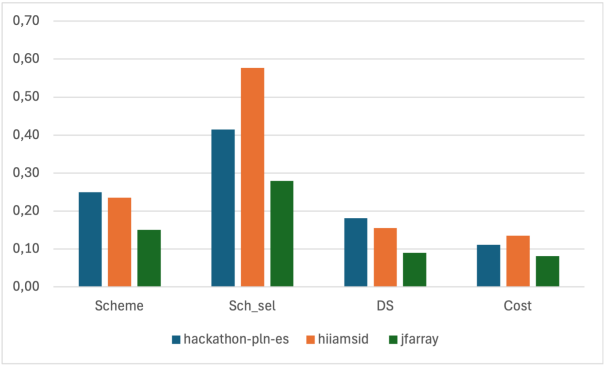


Figure 8. Precision@5 for the considered models. Graphical representation of the data in Table 2.

The next metric considered was the hit rate, HR, which is related to both the paragraph selected for system feedback (*selected*) and the presence of any relevant text that is useful for feedback (*relevant*). The HR is more meaningful than Prec@5 for the task considered here because it reflects the effectiveness of finding at least one adequate explanation, which is what is of interest in our context.

Table 3 shows the HR values for the different models when considering the presence of the selected text as an explanation for feedback (HR_SEL) and the presence of any valid paragraph as an explanation for feedback (HR_REL) as a hit. Figure 9 shows the corresponding graphical representation. As expected, the HR_SEL values were lower than the HR_REL values, which is a less demanding metric. However, the differences were not very marked, especially in the case of the best-performing models, *hackathon-pln-es* and *hiiamsid*.

Table 3. Hit rate (HR) and mean reciprocal rank (MRR) for the considered models. The HR is presented for both the selected text and any relevant text. The MRR is presented for the selected text.

Model	HR_SEL	HR_REL	MRR
hackathon-pln-es	0.719	0.802	0.610
hiiamsid	0.652	0.719	0.525
jfarray	0.462	0.635	0.343

However, all models exhibited considerably strong performance. The *hackathon-pln-es* model provided the selected paragraph, considered the most suitable, as an explanation in over 70% of cases, and in over 80% of cases, it provided a useful explanation. Even the model that ranked last, *jfarray*, obtained a useful explanation in over 60% of cases.

Figures 10 and 11 depict the HR considering the selected explanation (Figure 10) and any valid explanation (Figure 11) for the different topics related to the subject. Topics with fewer than five questions were discarded as the results were not considered meaningful.

Figure 9. Hit rate (HR) and mean reciprocal rank (MRR) for the considered models. Graphical representation of the data in Table 3.

We can observe that the results are similar in both charts, which are logically higher when considering the presence of any valid explanation as a hit. For most topics, the results were remarkable. This was the case for *schema selection*, *hash tables*, *connected components of a graph*, and *adjacency lists*. These are topics for which the questions usually have explanatory paragraphs in the reference texts. The lowest values corresponded to questions related to *computational cost*. These questions sometimes relate to problems that are not in the reference texts and, although the cost analysis process is similar to that of other problems that indeed appear, the final cost is different. The results for certain algorithmic schemes, such as the *backtracking*, *divide-and-conquer*, or *greedy* algorithms, were also lower than those for other topics. An analysis of the error cases in these topics showed that instances where no adequate explanations were found were frequently related to the *computational cost* of the algorithm under consideration.

Next, we considered the MRR. As mentioned above, it provides an approximation of the quality of the ranking of responses provided by each model. Instead of considering only whether relevant documents were retrieved, MRR considers the position of the first relevant document in the list of hits. Table 3 also includes the overall values of this metric for the different models (fourth column). We can see that for the *hackathon-pln-es* and *hiiamsid* models, the MRR exceeded 60% and 50%, respectively. This indicates that, on average, in more than 50% of cases, the system retrieved a relevant document in a relatively high position in the result list. Specifically, this means that, in more than 50% of the queries performed, the first relevant document retrieved appeared among the first results returned by the system.

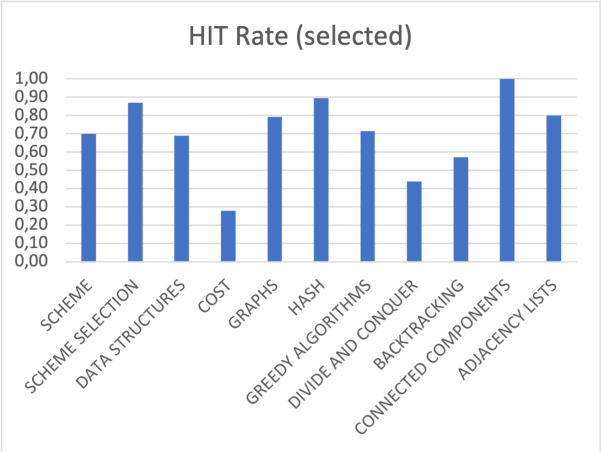


Figure 10. Hit rate (HR) considering the selection of a text as an explanation as a hit with the *hackathon-pln-es* model for different topics.

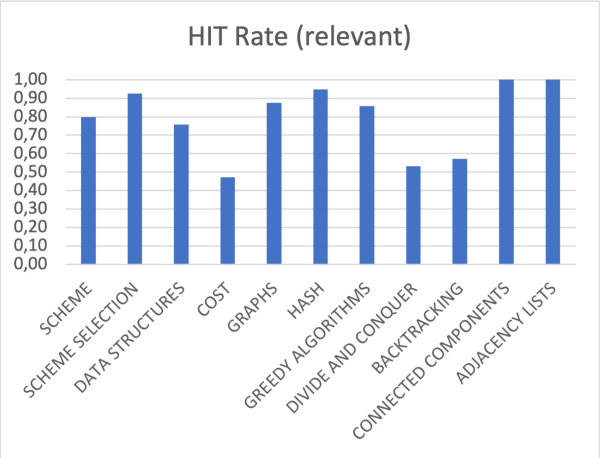


Figure 11. Hit rate (HR) considering the retrieval of a relevant text as an explanation as a hit with the *hackathon-pln-es* model for different topics.

Figure 12 shows the MRR results using the best model, *hackathon-pln-es*, for the different topics with more than five questions. We can see that the results are similar to those of the HR, although somewhat lower, since here, it was not only measured whether a relevant paragraph was found but also its position. As in the case of the HR, the results for some topics, such as *hash tables*, for which the MRR value exceeded 80%, stand out.

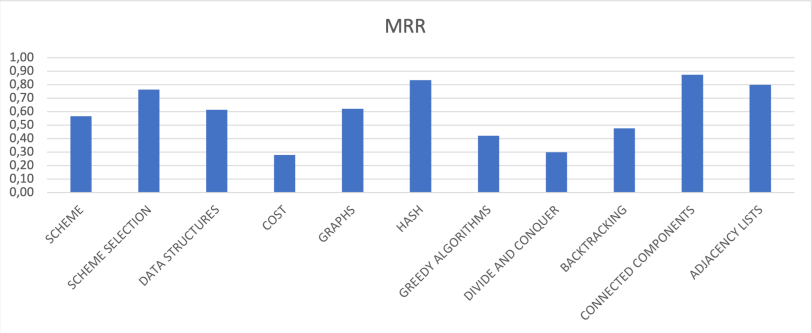


Figure 12. Mean reciprocal rank (MRR) considering the retrieval of a relevant text as an explanation with the *hackathon-pln-es* model for different topics.

Finally, Figure 13 shows the number of relevant paragraphs retrieved by each model for each position of the five considered, including the case of not having retrieved any. We can observe that the model *hackathon-pln-es* did not retrieve any relevant paragraphs in only 20% of cases. That is, in 80% of cases, it retrieved at least one useful paragraph as feedback. We can also see that the three models retrieved a relevant paragraph in the first position in a high percentage of cases: in about 80% of the cases for the *hiiamsid* model, about 80% for the *hackathon-pln-es* model, and about 75% for the *jfarray* model. The number of cases in which two relevant paragraphs were retrieved was logically lower but still high in 40% of cases for the *hackathon-pln-es* model. In about 20% of cases, three relevant paragraphs were retrieved by the *hiiamsid* and *hackathon-pln-es* models. In very few cases, four or five relevant paragraphs were retrieved by all models, which was expected since there were not many relevant paragraphs in the reference texts.

Considering the most representative metrics, HR and MRR, we observed that the *hackathon-pln-es* and *hiiamsid* models, which utilize larger vectors, delivered results that were reliable enough to use the chosen explanations as feedback. This is especially noteworthy when we define the hit rate as the inclusion of any relevant paragraph, even if it was not the exact one that was selected.

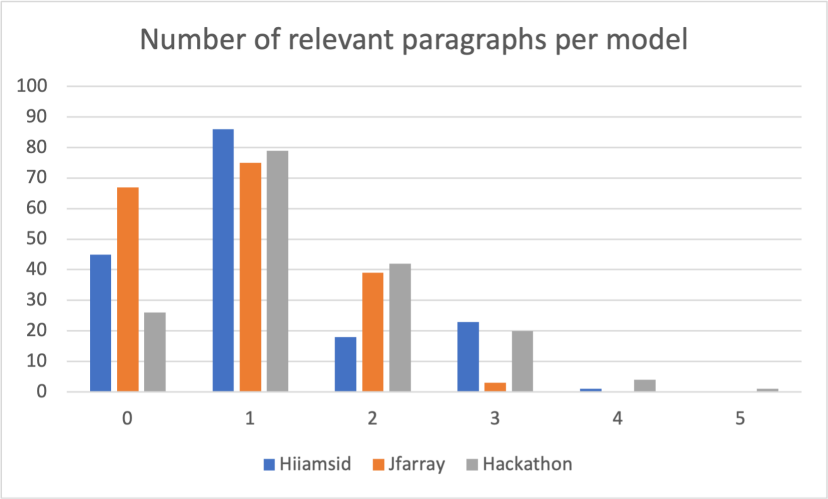


Figure 13. Number of questions for which each model retrieved a relevant paragraph as an explanation of the question at each position.

Furthermore, the results for both the HR and MRR were particularly high for certain topics, such as hash tables, connected components, and scheme selection, which involve specialized terminology. The poorest results were linked to questions about algorithmic cost, where the correct answer was sometimes missing from the reference text, and a similar one was insufficient.

4.2.1. Error Analysis

When analyzing the most frequent errors encountered during the evaluation, the first observation was that all the models encountered similar difficulties. The main source of errors was the lack of information in the reference texts about the specific algorithm to which a question referred. For example, consider the following question: *you have a vector, v, that stores integers in strictly increasing order, and you want to find out if there is any element that satisfies v[i] = i. What would be the most efficient strategy to solve the problem?*

The algorithm for this particular problem was not in the reference texts and the feedback generated consisted of examples of other problems that are solved using the same scheme in this problem, which is almost a *divide-and-conquer* problem.

Another common source of errors was questions related to cost. In some cases, these questions referred to problems for which the cost did not appear in the reference texts.

Another source of difficulty was questions referring to multiple topics rather than a specific one. In such cases, noise tended to degrade the results, as the best explanations were sometimes not among the first five selected paragraphs.

Finally, sometimes the explanations were less precise when trying to explain why an option was false, as the reference texts presented correct cases.

4.2.2. System Efficiency

The system demonstrated high efficiency in evaluating the semantic similarity between each of the 232 questions and a set of five reference paragraphs for each of them. A key advantage of this approach is that it does not require training, significantly reducing computational overhead. The execution times for the complete set of questions varied depending on the model used. Specifically, the *hiiamsid* model processed the dataset in 195.87 s, the *hackathon-pln-es* model in 120.87 s, and the *jfarray* model in 132.49 s. These results highlight the system's ability to perform large-scale similarity evaluations in a reasonable time frame, making it a practical solution for real-world applications.

4.3. Data on the Use of the Tool by Students

The tool was used during the 2023–2024 academic year by students in the course, spanning four months from November 2023 to February 2024, when the semester ended. As mentioned earlier, teaching was carried out online. The total number of students across the two computer science degrees where the tool was implemented was 291, with 85% male and 15% female students. There was also a diverse range of nationalities (Italy, Germany, Morocco, Romania, Cuba, and Colombia), although the majority, 95%, were Spanish. Due to data protection reasons, we do not have specific age data, but generally, the age of students at our university (across all degrees) is considerably higher than that of students attending in-person universities, with only one-third of them being under 30 years old.

Although the number of students enrolled in the course, including those of the two degrees in which it is taught, was 291, it should be taken into account that the dropout rate in distance learning is usually very high due to personal circumstances (work, family, etc.). In fact, only 142 students took the exam, among whom, with high probability, were those who actually used the tool.

It is common in many surveys that students with better academic performance are more likely to participate. However, despite being aware of the limitations and possible bias of the results, we consider that the responses collected provide valuable information that serves as a starting point for further improvement, although, of course, we will also try to extend the information collected to a larger number of students.

We verified that the tool had been used by 77 students, which means that more than 50% (specifically 54.2%) of students who actually took the course used the tool. Taking into account that, in many cases, students prepare for the course just before the exam, we consider that this is a high percentage of use.

Figure 14 shows a histogram of the number of students who answered a certain number of questions. We can see that 26 users made limited use of the tool, answering only between 0 and 9 questions. However, the remaining 51 users, representing more than 66% of all users, showed a good degree of adherence. Although the degree of use varied, a substantial (28, representing more than 36%) answered more than 50 out of the 107 questions contained in the tool.

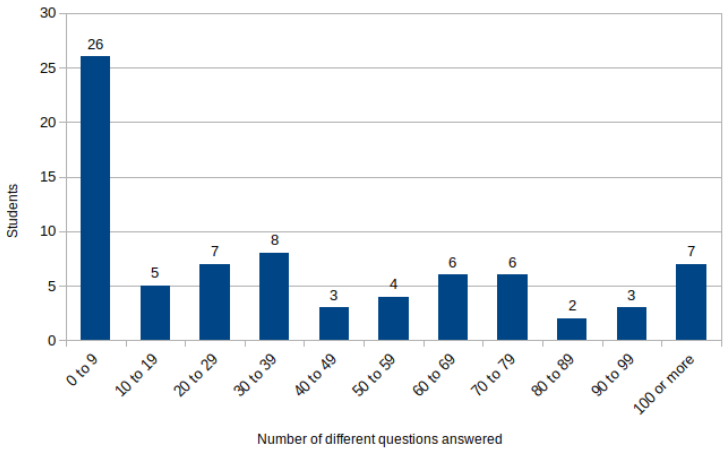


Figure 14. Histogram of the number of students who answered N questions.

4.4. Students' Perceptions of the Tool

To gauge student perception, we conducted a survey consisting of seven questions, as detailed in Appendix A. Figures 15–20 present the results of the anonymous questionnaire completed by the students.

Nineteen students participated in the survey, representing approximately 25% of students who used the tool (77, as explained above).

The first aspect considered was the overall utility of the tool. As shown in Figure 15, the majority of students found the tool very or quite useful, with no students indicating that they found it unhelpful. Similarly, virtually all students considered the tool easy to use, as shown in the results presented in Figure 16.

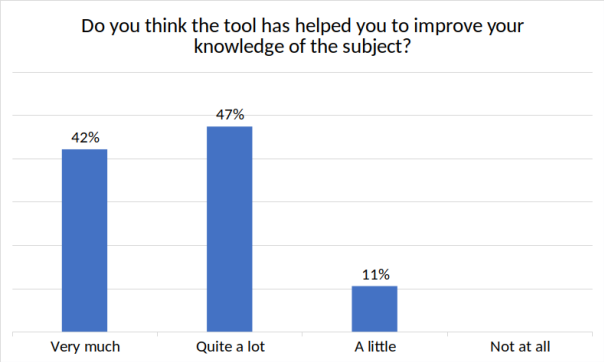


Figure 15. Utility of the tool as perceived by students.

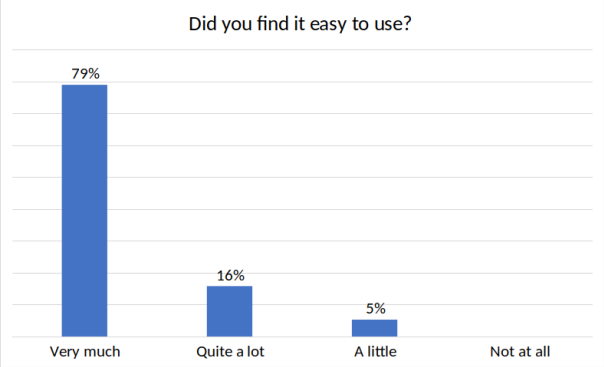


Figure 16. Ease of use of the tool as perceived by students.

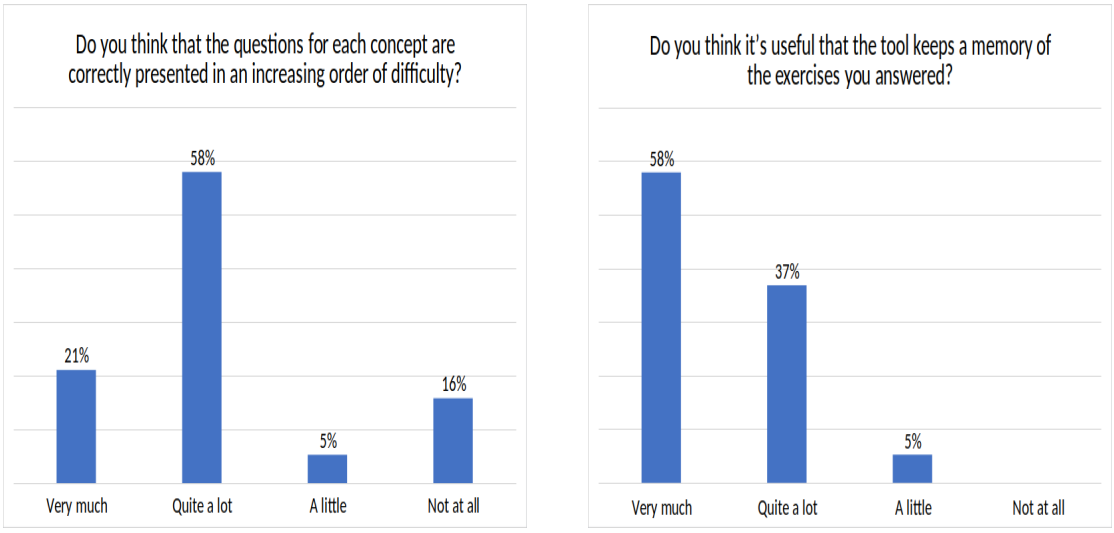
Another aspect students were asked to rate was the effectiveness of the personalization mechanisms, including the incremental difficulty of the questions and the recording of usage history. Figure 17 shows the results. Regarding whether questions on a specific topic were presented in increasing order of difficulty (Figure 17a), there was more disparity in opinions, as some studies have shown that students' perceptions do not always align with actual results [52]. In our case, the difficulty associated with each question was objectively calculated based on students' relative successes in exams from previous courses where the question had been used. We also inquired about the usefulness of the tool, presenting questions in a personalized manner and avoiding repetition of questions already answered correctly by students, which required maintaining a usage record for each user. In this case, as shown in Figure 17b, there was a high level of agreement among students who found it very useful. We can see that in terms of the usage record, opinions were much more uniform than those related to the difficulty of the questions.

Regarding the concepts for which the tool proved most useful, we can see in Figure 18 that there was a wide range of opinions, although aspects related to computational cost and graphs stand out.

We also inquired about the usefulness of the explanations regarding answer corrections, and the results are shown in Figure 19. The majority of students found it very useful (58%) or quite useful (32%), with only 11% considering it of limited usefulness.

Finally, we queried students about the type of questions for which explanations are most useful, and the results are presented in Figure 20. It should be noted that for this question, students could select more than one response. We observed that, for general topics, questions

related to computational cost calculation stood out, whereas for questions related to data structures and algorithmic schemes, the results were evenly distributed. Similarly, we observed that the results were the same for both theoretical and practical questions. Additionally, students found explanations for incorrect answers particularly useful, as expected.



(**a**) (**b**)

Figure 17. Evaluation of personalization mechanisms. (**a**) Appropriateness of the increasing order of difficulty of the questions for each topic. (**b**) Usefulness of recording the usage history of the tool by each student.

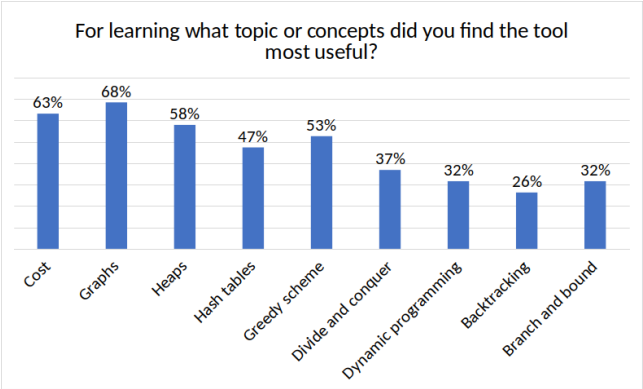


Figure 18. Topics or concepts for which the tool is most useful.

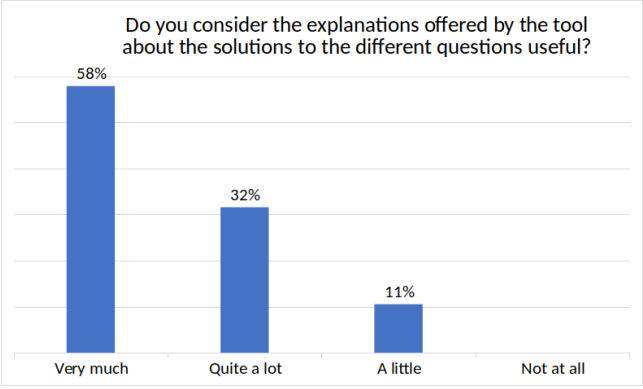


Figure 19. Usefulness of the explanations provided by the tool in terms of the correctness of the answers.

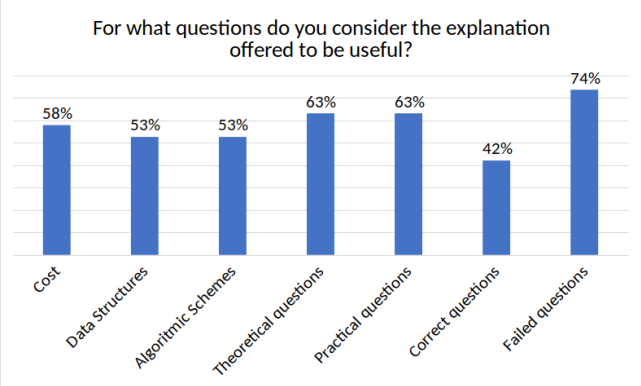


Figure 20. Types of questions for which explanations are most useful.

5. Discussion

Self-assessment tools are integral to modern education, offering students valuable insights into their knowledge gaps and facilitating their learning journeys. Feedback plays a pivotal role in this process, guiding learners in real time to identify areas for improvement and adjust their learning strategies accordingly. While traditional feedback methods focus on correctness, supplementary instructional feedback provides deeper insights into learners' misconceptions and the rationale behind correct answers.

Technological advancements, particularly in natural language processing (and artificial intelligence in general), present promising opportunities for enhancing educational tools. By automating the generation of explanations tailored to specific subjects, these advancements enable more personalized and effective feedback mechanisms. Our proposed self-assessment tool, designed for an advanced computer science course, offers detailed explanations for both correct and incorrect responses.

While LLMs, such as GPT, have demonstrated remarkable capabilities in generating text across different domains, their use for generating feedback automatically without supervision presents significant challenges. One primary concern is the potential inclusion of erroneous or misleading responses. Language models operate by learning patterns from vast datasets, which may contain inaccuracies or biased information. In the context of providing feedback, this poses a risk of generating responses that are actually incorrect or inappropriate. Moreover, feedback often requires contextual understanding and domain-specific knowledge, aspects that may be beyond the capabilities of current language models. Therefore, utilizing LLMs for automatically generating feedback without supervision is not advisable, highlighting the necessity of human oversight to ensure quality and relevance in this process. Therefore, to leverage the most advanced language representation tools while ensuring that the explanations provided to students are highly reliable, we used semantic similarity techniques to extract the most appropriate paragraphs from a text as explanations for a given question.

Our exploration of different neural language models for automatically generating explanations revealed significant differences in performance. The *hackathon-pln-es* model consistently outperformed others, followed closely by the *hiiamsid* model. Both models share a larger representation than the third model. These variations in performance highlight the importance of model selection in achieving accurate and relevant feedback generation.

Furthermore, we found that the nature of the questions greatly influences the results. Questions related to algorithmic schemes and data structures yielded higher precision and recall rates than those concerning computational cost. This discrepancy underscores the challenge of retrieving relevant explanations for more nuanced or specialized topics within the field of computer science.

We also surveyed students who had used the tool to gauge the usefulness of various aspects. Overall, the results indicated that students highly valued the different features we developed, especially the customization of the proposed questions and the provision of explanations for answer correctness. In particular, students found the explanations very helpful when they encountered failed questions.

One limitation of our study is its focus solely on theoretical issues, which make up approximately 50% of the questions. Nowadays, it is feasible to leverage large language models like ChatGPT to generate explanations for any question type. However, this would need to be accompanied by suitable verification mechanisms to ensure the accuracy of the feedback provided.

In our approach, we employed a controlled deployment of some of the latest language models to select paragraphs that can serve as explanations. These explanations are not erroneous as they originated from a textbook on the subject matter that has been validated by the teaching team.

6. Conclusions

In this work, we developed a system that generates explanations for questionnaire responses by selecting relevant paragraphs from reliable sources that provide justifications for the answers. The system is based on semantic similarity techniques that use large language models (LLMs) to represent the texts whose similarities are being compared. Since the explanations are drawn from reference texts, the risk of generating incorrect explanations is reduced, which can occur with other generative models. The system, which focuses on theoretical questions, performs an additional classification of the types of questions presented in the given environment to conduct more refined searches for similar paragraphs. The evaluation of the relevance of the selected explanations shows that some models provide an adequate explanation in up to 80% of cases.

The key contributions of our work include the development of a methodology for automatically generating explanations by leveraging neural language models to select relevant paragraphs from available documents. We conducted extensive evaluations of the different models, culminating in the integration of the explanations into our self-assessment tool, thereby enhancing feedback for students. Notably, our evaluation metrics demonstrated the effectiveness of these models in retrieving relevant explanations.

Future Work

We plan to continue advancing along these lines, improving the automatic generation of explanations from different perspectives.

The main limitations of the model relate to its dependence on reference texts and the exclusion of practical questions in the process of generating explanations, and we intend to make progress in overcoming these limitations.

In the process of generating explanations through semantic similarity, the selection of reference texts closely related to the given question is crucial to ensure the relevance and accuracy of the explanation provided. If the texts retrieved are too general or contain irrelevant information, noise is introduced into the process, which negatively impacts the clarity and usefulness of the answer given to the learner. To mitigate this problem, it is essential to employ information retrieval strategies that prioritize reliable and contextually relevant sources. There are several strategies for retrieving appropriate texts. For example, in specific contexts, as in our case, Augmented Retrieval Generation (RAG) techniques can be used, combining linguistic models such as BERT or RoBERTa with efficient search mechanisms such as BM25 or FAISS to locate textual fragments highly relevant to the given question. In the future, we intend to explore these possibilities.

Another line of improvement we will explore is tuning the models with specific texts on algorithms and data structures.

We also want to explore other ways of employing large ChatGPT-type language models, which would also allow us to address practical issues. The responses of these generative models are not guaranteed to be reliable, so we would check them with natural language processing techniques and accredited reference texts, such as those used in this work. In addition, we also want to advance the personalization of the tool, allowing users, for example, to select whether they want the explanations summarized, detailed, or presented with simple texts.

Other lines of research that we intend to explore include the integration of the tool in LMSs like Moodle, the improvement of the presentation of explanations by including resources such as images and animations, and the extension of the tool to other languages.

We also intend to validate it by generating self-assessment tools for other subjects. This will help us test the tool on a larger population of students.

Author Contributions: Conceptualization, L.A.; Methodology, L.A., F.L.-O., L.P. and J.M.-R.; Software, L.A., F.L.-O., L.P. and J.M.-R.; Validation, L.A., F.L.-O., L.P. and J.M.-R.; Writing – original draft, L.A.; Writing—review and editing, F.L.-O., L.P. and J.M.-R.; Visualization, F.L.-O.; Funding acquisition, L.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the UNED 2024 project for the innovation group INEDA (GID2017-1) and the Spanish Ministry of Science and Innovation within the OBSER-MENH project (MCIN/AEI/10.13039 and NextGenerationEU"/PRTR) under grant TED2021-130398B-C21 and the EDHER-MED Project under grant PID2022-136522OB-C21. It was also financed by the Spanish Ministry of Science and Innovation (FairTransNLP project (PID2021-124361OB-C32)), funded by MCIN/AEI/10.13039/501100011033 and ERDF, EU A way of making Europe.

Data Availability Statement: Data will be made available on request.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Self-Assessment Tool Questionnaire

The questionnaire includes the following questions:

1. Do you think the tool has helped you improve your knowledge of the subject? (Very much | Quite a lot | A little | Not at all).
2. Did you find it easy to use? (Very much | Quite a lot | A little | Not at all).
3. Do you think that the questions for each concept are correctly presented in increasing order of difficulty? (Very much | Quite a lot | A little | Not at all).
4. Do you think it's useful that the tool keeps a memory of the exercises you answered? (Very much | Quite a lot | A little | Not at all).
5. For learning what topics or concepts did you find the tool most useful? (Cost | Graphs | Heaps | Hash tables | Greedy scheme | Divide and conquer | Dynamic programming | Backtracking | Branch and bound).
6. Do you consider the explanations offered by the tool about the solutions to the different questions useful? (Very much | Quite a lot | A little | Not at all).
7. For what questions do you consider the explanation offered helpful? (Cost | Data structures | Algorithmic schemes | Theoretical questions | Practical questions | Correct questions | Failed questions).
8. Any comments or suggestions you would like to share with us to help us improve the application?

Appendix B. Examples of Explanations Generated for a Specific Question

Here are some examples of the answers provided by the models to a specific question.

Let us consider the language model *hiiamsid* and the following question related to the *branch-and-bound* algorithm, which is TRUE: *During the execution of a branch-and-bound algorithm, a solution is found that is better than the best solution existing at that time and that improves the optimistic estimate of the top of the mound. This implies that the algorithm has definitely found the solution.*

Five paragraphs are obtained, and the first two are as follows:

SIMILARITY = 0.678

We can write the general branch-and-bound scheme for a minimization problem as follows: The algorithm accumulates in a heap the elements that, once completed, can be solutions to the problem. In the heap, they are kept sorted by an optimistic estimate...

RELEVANT

SIMILARITY = 0.611

Branch and bound is also a scheme for exploring an implicit directed graph. In this scheme, the nodes are not explored following the sequence in which they have been generated, as done in the backtracking scheme, but the function to be optimized is used to establish preferences between the nodes to be explored. The path is now directed by the most promising active node, so that...

RELEVANT

SELECTED

The *hackathon-pln-en* model also selects the same paragraphs, although the associated similarity values are different, with 0.678 for the first paragraph and 0.611 for the second one. In the case of this question, the same is true for the remaining model, *jfarray*.

The specific similarity values are not taken into account (they are not comparable across different models), except for selecting the five most relevant within each model and establishing an order among them. In the previous example, both paragraphs are relevant and could be useful as feedback. However, the second one was selected as it provides a clearer explanation of why the statement is TRUE. However, the first would also have been perfectly valid.

References

1. Fierro-Saltos, W.; Sanz, C.; Zangara, A.; Guevara, C.; Arias-Flores, H.; Castillo-Salazar, D.; Varela-Aldás, J.; Borja-Galeas, C.; Rivera, R.; Hidalgo-Guijarro, J.; et al. Autonomous learning mediated by digital technology processes in higher education: A systematic review. In Proceedings of the Human Systems Engineering and Design II: Proceedings of the 2nd International Conference on Human Systems Engineering and Design (IHSED2019): Future Trends and Applications, Universität der Bundeswehr München, Munich, Germany, 16–18 September 2019; Springer: Berlin/Heidelberg, Germany, 2020; pp. 65–71.
2. Yan, Z. Self-assessment in the process of self-regulated learning and its relationship with academic achievement. *Assess. Eval. High. Educ.* **2020**, *45*, 224–238. [CrossRef]
3. Yan, Z.; Carless, D. Self-assessment is about more than self: The enabling role of feedback literacy. *Assess. Eval. High. Educ.* **2022**, *47*, 1116–1128. [CrossRef]
4. Andrade, H.L. A critical review of research on student self-assessment. In *Frontiers in Education*; Frontiers Media SA: Lausanne, Switzerland, 2019; Volume 4, p. 87.
5. Guzmán, E.; Conejo, R.; Pérez-de-la Cruz, J.L. Improving student performance using self-assessment tests. *IEEE Intell. Syst.* **2007**, *22*, 46–52. [CrossRef]
6. Benchoff, D.E.; Gonzalez, M.P.; Huapaya, C.R. Personalization of tests for formative self-assessment. *IEEE Rev. Iberoam. Tecnol. Aprendiz.* **2018**, *13*, 70–74. [CrossRef]
7. Shute, V.J. Focus on formative feedback. *Rev. Educ. Res.* **2008**, *78*, 153–189. [CrossRef]
8. Panadero, E.; Lipnevich, A.A. A review of feedback models and typologies: Towards an integrative model of feedback elements. *Educ. Res. Rev.* **2022**, *35*, 100416. [CrossRef]
9. Yan, Z.; Panadero, E.; Wang, X.; Zhan, Y. A systematic review on students' perceptions of self-assessment: Usefulness and factors influencing implementation. *Educ. Psychol. Rev.* **2023**, *35*, 81. [CrossRef]

10. Ifenthaler, D.; Schumacher, C.; Kuzilek, J. Investigating students' use of self-assessments in higher education using learning analytics. *J. Comput. Assist. Learn.* **2023**, *39*, 255–268. [CrossRef]
11. Barana, A.; Boetti, G.; Marchisio, M. Self-Assessment in the Development of Mathematical Problem-Solving Skills. *Educ. Sci.* **2022**, *12*, 81. [CrossRef]
12. Ross, M.; Litzler, E.; Lopez, J. Meeting students where they are: A virtual computer science education research (CSER) experience for undergraduates (REU). In Proceedings of the 52nd ACM Technical Symposium on Computer Science Education, Virtual, 13–20 March 2021; pp. 309–314.
13. El-Maaddawy, T. Enhancing learning of engineering students through self-assessment. In Proceedings of the 2017 IEEE Global Engineering Education Conference (EDUCON), Athens, Greece, 26–28 April 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 86–91.
14. Hao, Q.; Smith IV, D.H.; Ding, L.; Ko, A.; Ottaway, C.; Wilson, J.; Arakawa, K.H.; Turcan, A.; Poehlman, T.; Greer, T. Towards understanding the effective design of automated formative feedback for programming assignments. *Comput. Sci. Educ.* **2022**, *32*, 105–127. [CrossRef]
15. Olsen, T.; Hunnes, J. Improving students' learning–=The role of formative feedback: experiences from a crash course for business students in academic writing. *Assess. Eval. High. Educ.* **2024**, *49*, 129–141. [CrossRef]
16. Lipnevich, A.A.; Panadero, E. A review of feedback models and theories: Descriptions, definitions, and conclusions. In *Frontiers in Education*; Frontiers: Lausanne, Switzerland, 2021, Volume 6, p. 720195.
17. Spady, R.; Karge, B.D. The Value of Formative Feedback in Graduate Online Courses. *Distance Learn.* **2022**, *19*, 73–82.
18. Gálvez-López, E. Formative feedback in a multicultural classroom: A review. *Teach. High. Educ.* **2025**, *30*, 463–482. [CrossRef]
19. Serral, E.; Snoeck, M. Conceptual framework for feedback automation in SLEs. In *Smart Education and E-Learning*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 97–107.
20. Deeva, G.; Bogdanova, D.; Serral, E.; Snoeck, M.; De Weerdt, J. A review of automated feedback systems for learners: Classification framework, challenges and opportunities. *Comput. Educ.* **2021**, *162*, 104094. [CrossRef]
21. Gibbs, G.; Simpson, C. Conditions under which assessment supports students' learning. *Learn. Teach. High. Educ.* **2005**, 3–31.
22. Glover, C.; Brown, E. Written feedback for students: Too much, too detailed or too incomprehensible to be effective? *Biosci. Educ.* **2006**, *7*, 1–16. [CrossRef]
23. Voelkel, S.; Varga-Atkins, T.; Mello, L.V. Students tell us what good written feedback looks like. *FEBS Open Bio* **2020**, *10*, 692–706. [CrossRef]
24. Cavalcanti, A.P.; Barbosa, A.; Carvalho, R.; Freitas, F.; Tsai, Y.S.; Gašević, D.; Mello, R.F. Automatic feedback in online learning environments: A systematic literature review. *Comput. Educ. Artif. Intell.* **2021**, *2*, 100027. [CrossRef]
25. Trausan-Matu, S.; Dascalu, M.; Rebedea, T. PolyCAFe—Automatic support for the polyphonic analysis of CSCL chats. *Int. J. Comput.-Support. Collab. Learn.* **2014**, *9*, 127–156. [CrossRef]
26. Ono, Y.; Ishihara, M.; Yamashiro, M. Preliminary construction of instant qualitative feedback system in foreign language teaching. In Proceedings of the 2013 Second IIAI International Conference on Advanced Applied Informatics, Los Alamitos, CA, USA, 31 August–4 September 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 178–182.
27. Keuning, H.; Jeuring, J.; Heeren, B. A systematic literature review of automated feedback generation for programming exercises. *ACM Trans. Comput. Educ. (TOCE)* **2018**, *19*, 1–43. [CrossRef]
28. Duong, T.N.B.; Shar, L.K.; Shankararaman, V. AP-Coach: Formative feedback generation for learning introductory programming concepts. In Proceedings of the 2022 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), Hung Hom, Hong Kong, 4–7 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 323–330.
29. Demaidi, M.N.; Gaber, M.M.; Filer, N. OntoPeFeGe: Ontology-based personalized feedback generator. *IEEE Access* **2018**, *6*, 31644–31664. [CrossRef]
30. Demaidi, M.N. Ontology Validation & Utilisation For Personalised Feedback In Education. Ph.D. Thesis, Birmingham City University, Birmingham, UK, 2018.
31. Chang, M.; D'Aniello, G.; Gaeta, M.; Orciuoli, F.; Sampson, D.; Simonelli, C. Building ontology-driven tutoring models for intelligent tutoring systems using data mining. *IEEE Access* **2020**, *8*, 48151–48162. [CrossRef]
32. Araujo, L.; Martinez-Romo, J.; Plaza, L.; López-Ostenero, F. Analysis of the Generation of Explanations for Self-assessment Exercises on Algorithm Schemes and Data Structures. In Proceedings of the INTED2023 Proceedings, Valencia, Spain, 6–8 March 2023; IATED: 2023; pp. 1554–1563.
33. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
34. Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; Alsaadi, F.E. A survey of deep neural network architectures and their applications. *Neurocomputing* **2017**, *234*, 11–26. [CrossRef]
35. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
36. Corley, C.D.; Mihalcea, R. Measuring the semantic similarity of texts. In Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, MI, USA, 30 June 2005; pp. 13–18.

37. Fellbaum, C. WordNet. In *Theory and Applications of Ontology: Computer Applications*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 231–243.
38. Amur, Z.H.; Kwang Hooi, Y.; Bhanbhro, H.; Dahri, K.; Soomro, G.M. Short-Text Semantic Similarity (STSS): Techniques, Challenges and Future Perspectives. *Appl. Sci.* **2023**, *13*, 3911. [CrossRef]
39. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z. Dbpedia: A nucleus for a web of open data. In Proceedings of the International Semantic Web Conference, Busan, Republic of Korea, 11–15 November 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 722–735.
40. Rozeva, A.; Zerkova, S. Assessing semantic similarity of texts–methods and algorithms. In Proceedings of the AIP Conference Proceeding, Provo, UT, USA, 16–21 July 2017; AIP Publishing: 2017; Volume 1910.
41. Majumder, G.; Pakray, P.; Gelbukh, A.; Pinto, D. Semantic textual similarity methods, tools, and applications: A survey. *Comput. Y Sist.* **2016**, *20*, 647–665. [CrossRef]
42. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
43. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
44. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
45. Prakoso, D.W.; Abdi, A.; Amrit, C. Short text similarity measurement methods: A review. *Soft Comput.* **2021**, *25*, 4699–4723. [CrossRef]
46. Araujo, L.; López-Ostenero, F.; Martínez-Romo, J.; Plaza, L. Deep-Learning Approach to Educational Text Mining and Application to the Analysis of Topics' Difficulty. *IEEE Access* **2020**, *8*, 218002–218014. [CrossRef]
47. López-Ostenero, F.; Plaza, L.; Araujo, L.; Martínez-Romo, J. Self-Assesment tool with topic-driven navigation for algorithms learning. In Proceedings of the IEEE Global Engineering Education Conference, EDUCON 2022, Tunis, Tunisia, 28–31 March 2022; Kallel, I., Kammoun, H.M., Hsairi, L., Eds.; IEEE: Piscataway, NJ, USA, 2022; pp. 356–363.
48. López-Ostenero, F.; Martínez-Romo, J.; Plaza, L.; Araujo, L. Personalized Self-Assessment Tool Using a Telegram Bot: A Case Study on Data Structures and Algorithms. In Proceedings of the IEEE Global Engineering Education Conference, EDUCON Kos, Greece, 8–11 May 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–8.
49. Cañete, J.; Chaperon, G.; Fuentes, R.; Ho, J.H.; Kang, H.; Pérez, J. Spanish pre-trained bert model and evaluation data. *arXiv* **2023**, arXiv:2308.02976.
50. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv* **2020**, arXiv:1910.03771.
51. Wu SJ, D.M. The surprising cross-lingual effectiveness of BERT. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 833–844.
52. van de Watering, G.; van der Rijt, J. Teachers' and students' perceptions of assessments: A review and a study into the ability and accuracy of estimating the difficulty levels of assessment items. *Educ. Res. Rev.* **2006**, *1*, 133–147. [CrossRef]