Benchmarking LLMs for Content Expansion: Measuring Novelty in Iterative Course Outline Generation

Vickey Ghimire, Jaljala Shrestha Lama, Bijay Dhungana, Nazmus Sadat, Nicholas Caporusso
School of Computing and Analytics
Northern Kentucky University
Highland Heights, KY, United States
{sadatm1, caporusson1}@nku.edu

***Abstract*—Iterative prompting is a technique commonly used to progressively refine the output produced by Large Language Models (LLMs) to obtain more comprehensive content. This method is particularly useful in the context of education for drafting and refining lists of topics to be included in course outlines, syllabi, and lecture plans. However, one of the main challenges in iterative prompting is determining whether, at each iteration, the model outputs truly new material or merely rephrases existing content. While iterative refinement is a common practice, it often results in repetition, which requires educators to conduct time-consuming manual checks, thereby limiting the benefits provided by LLM-assisted course design. To mitigate this, we introduce a framework that applies a novelty metric, integrating both lexical and semantic similarity measures, to automatically categorize generated outputs as novel, redundant, or requiring human judgment. We tested this framework across seven widely used LLMs (ChatGPT o3, Claude 4 Sonnet, Gemini 2.5 Pro, DeepSeek R1, Grok 3, Qwen 2.5, and Llama 3.3) to create a Java programming course outline over multiple iterations. Results from conversations containing over 23,751 individual topics showed that DeepSeek R1 produced the highest percentage of novel content. Moreover, our method successfully automated classification for 96.7% of the outputs, substantially reducing the proportion of items that require manual review.**

***Index Terms*—Large language models, Iterative Prompting, Course Outline, Educational Content Design, Instructional design, Pedagogy.**

I. Introduction

Recent progress in Large Language Models (LLMs) has transformed generative AI systems into everyday tools that can be seamlessly integrated into content creation tasks, including the development of resources for teaching and learning. An expanding body of research investigates how LLMs can enhance the generation of materials that instructors can use in their courses [1]–[4]. Prior studies [5] highlight the value of LLMs in aiding instructional design, particularly in supporting the creation of structured academic resources, such as course outlines, syllabi, assignments, and tailored learning pathways. Thanks to their knowledge base and capacity to interpret context and simulate human reasoning, these models can help instructors draft course outlines, analyze existing curricular material, and propose optimized course structures, automating repetitive authoring tasks and suggesting potential improvements that would otherwise require additional research and effort. As a result, instructors are increasingly adopting LLMs as collaborators in designing syllabi and course structures. In a typical course outline generation scenario, the LLM would produce an initial draft that the instructor can evaluate and optimize based on the specific objectives of the course. The support offered by LLMs, in turn, enables educators to devote more time to improving course delivery, advancing learning objectives, and providing students with personalized mentorship.

The ability of LLMs to generate structured course content is particularly beneficial in dynamic fields where educational resources can quickly become outdated within a single academic cycle. For instance, in Computer Science, courses focusing on software engineering, data science, software development, artificial intelligence, and cybersecurity require constant updates to reflect newly released frameworks, evolving industry standards, and emerging research breakthroughs. In such contexts, instructors can address the inherent complexity of course design by using LLMs to integrate the latest knowledge into course modules, propose updated examples and case studies, and align learning objectives with current professional practices. To this end, iterative prompting can be utilized as a technique to refine the initial, high-level draft produced by an LLM and gradually shape it into a more detailed structure.

The iterative prompting-based refinement process involves prompting the LLM several times to direct the model to incorporate additional material, integrate emerging topics, sharpen learning outcomes, or increase the applied focus of the course. Each time, the LLM would respond with a new and refined course outline. A key issue, however, lies in how various LLMs generate content and respond to such iterative interactions. Specifically, the challenge is understanding whether, in each iteration, they genuinely broaden the scope of the outline with new information or simply rephrase what has already been presented. Although educators can manually evaluate successive outputs to extract and merge novel content, doing so introduces a trade-off: the convenience of LLM-generated revisions versus the time and effort required to assess the novelty and relevance of the content. This challenge becomes

even more significant when the initial prompt already contains a partially developed outline, making it harder to discern whether subsequent revisions genuinely contribute new value or merely reorganize existing material, and requiring additional effort from the instructor.

In this paper, we present a novel approach that measures the novelty of LLM output in the context of iterative prompting. In the context of course outline generation, the proposed method aims at automatically labeling the output produced by an LLM as either new or repeated (or rephrased), making it easier for the instructor to evaluate which content to consider. Indeed, our work recognizes that curriculum design is not only a matter of content similarity, but also a process of aligning learning objectives, instructional activities, and assessments [6]–[8]. Therefore, our contribution is a tool designed to surface candidate novelties for educator judgment, thereby helping experts review the LLM output to verify alignment, appropriateness, and coherence. While decisions about whether a topic is relevant often depend on course-specific goals and the instructor's professional perspective, and therefore call for contextual consideration and human review, the question of whether content is genuinely new in iterative prompting can be assessed automatically. Such novelty detection can streamline revision by directing instructors' attention only to the areas that warrant closer evaluation. Additionally, content novelty measures might be utilized to help instructors evaluate different prompting strategies.

Finally, the proposed method enables comparing the behavior and performance of different LLMs in response to iterative prompting in various other applications and fields, thus providing a benchmark for evaluating model families and temperature settings based on specific tasks. This, in turn, could ultimately feed back into the architecture of LLMs to improve reasoning techniques beyond authoring tools for educators.

II. Related Work

Several studies have explored the generation of educational content driven by LLMs, including curriculum, lesson plans, and assessment items. However, far less attention has been devoted to understanding how different prompting strategies, and in particular iterative prompting, affect the evolution and quality of the generated content.

Sridhar et al. [2] demonstrated that GPT-4 can draft learning objectives according to Bloom's taxonomy, enabling instructors to create syllabi in 50% less time. However, the study also noted variability in objective specificity between disciplines. Sajja et al. [9] developed an AI-augmented intelligent educational framework that automatically generates course-specific intelligent assistants for different disciplines and academic levels. Fan et al. [10] developed an interactive LLM-based system to assist new teachers in preparing lesson plans, including a structured outline, materials, and examples. Yan et al. [5] surveyed 118 studies and showed that GPT models effectively produce course outlines and quiz questions. However, the results often require manual editing to ensure domain precision and pedagogical alignment.

Attard and Dingli [11] developed a comprehensive system utilizing GPT-4 to automate educational content generation within Intelligent Tutoring Systems. Their methodology employed prompt engineering and fine-tuning with domain-specific datasets aligned with Bloom's Taxonomy. The system incorporated content validation mechanisms, including human-in-the-loop processes to filter inaccuracies. The multi-LLM agents framework by Solaiman et al. [12] utilized the Crew AI framework to coordinate multiple autonomous agents for generating educational content targeted at children under 13 years old. Each agent had a specific role in content generation, and this collaborative approach performed better than single-model implementations. Hu et al. [13] proposed a prompt-based pipeline where LLMs simulate a teaching session, generate reflective feedback, and iteratively refine lesson plans. Unlike works that position LLMs primarily as tutors for students, this method focused on supporting teachers. Evaluation of high school mathematics lessons showed that LLM-refined plans improve over baseline LLM outputs and, in some cases, match or surpass those of experienced teachers.

Dornburg and Davin [14] investigated the effectiveness of ChatGPT in developing lesson plans for foreign language instruction, focusing on how variations in prompt specificity influence the quality and consistency of the generated content. Their findings revealed that providing additional context does not necessarily lead to improved outcomes, and in some cases, identical prompts produced markedly different results. In another study, Bao et al. [15] proposed a framework for iteratively generating and refining explanations for multiple-choice questions. Their results demonstrated that repeated prompting enhanced the clarity and effectiveness of the explanations. Similarly, Zheng et al. [4] introduced a lesson-plan generation framework where an LLM applies self-critique based on educator-defined evaluation criteria before refining its output. Evaluations conducted by experienced teachers revealed that this approach yielded higher-quality lesson plans compared to both alternative LLM-based methods and human-authored plans.

Beyond lesson planning, several studies have compared LLMs head-to-head in educational text generation tasks. For instance, Ashrafimoghari et al. [16] examined seven models on official GMAT problems, reporting GPT-4 Turbo as the most accurate, followed by Claude 2.1 and Gemini 1.0 Pro. Likewise, Pack et al. [17] evaluated GPT-4, Claude 2, PaLM 2, and GPT-3.5 on automated essay scoring for English language learners, finding GPT-4 to have the most substantial alignment with human ratings and the highest intra-rater reliability. Together, these findings highlight persistent reliability issues in different large language models.

Although this body of work has advanced understanding of the LLM capabilities in educational contexts, no prior research has systematically examined the originality of LLM-generated content for course outline development. Our study addresses this gap by analyzing the responses of seven different LLMs

to identical iterative prompts and applying a comparison algorithm to evaluate both the novelty of generated content and the consistency of response within each model.

III. Assessing Novelty in Iterative Prompting

The ultimate goal of our work is to design and develop systems that automatically assess the novelty of the output of an LLM tasked with producing an outline based on a structured or unstructured knowledge base. By doing this, a human evaluator can leverage the automatic assessment to iteratively prompt the LLM to refine its output and explore its knowledge base (whether internal or supported by Retrieval Augmented Generation), generating a progressively rich, detailed, and more defined outline. This work has applications in various fields, but is particularly relevant in education, where LLMs are utilized to generate course materials, including course outlines, syllabi, or lesson plans, from sources such as research papers, case studies, and lecture notes. Although the output would still require the instructor's review, automating the content generation process to expand an outline in terms of breadth and scope could simplify educators' work by providing them with rich, structured drafts. Ultimately, educators already use iterative prompting as a technique for revising course materials, particularly course outlines, and expanding them horizontally (i.e., by introducing additional relevant topics) and vertically (i.e., by including materials that help expand the knowledge of a specific topic). In this context, the role of our system is to assist instructors by identifying potentially valuable new material and highlighting redundancies, rather than replacing the expert review and completely automating the pedagogical judgment that determines what belongs in a curriculum.

To this end, in our previous work [18], we proposed a "novelty metric" (NOV) that evaluates whether LLMs truly add meaningful and new content when prompted repeatedly, or if they mostly repeat or rephrase information from earlier outputs, answering the question "How can we measure how much new information an LLM produces when prompted to refine and expand its response?" In this paper, we provide an extensive application of the proposed novelty metric to the scenario of an instructor asking an LLM to define the outline of a course. After the first output (i.e., an initial list of topics), the response of the LLM to subsequent prompts, iteratively asking for more detailed information, will consist of a new list where each topic can be divided into the following groups:

- *Exactly the same content:* The topic was already present in a previous version and, therefore, can be disregarded because the instructor has already evaluated it.
- *Completely different content:* Novel contributions from the latest response that expand and help explore a topic horizontally or vertically, or introduce new topics. These should be highlighted for instructor review and considered for inclusion based on their relevance for the course.
- *Similar content:* This includes repetitions, paraphrased versions of previously introduced items, while incorporating some new information not presented in the previous iteration(s). The issue with this type of content is that it can be considered either as similar to or different from previously seen output. This material, typically located at the center of the similarity spectrum, involves potentially overlapping entries that require further analysis. In such cases, additional effort from the instructor is needed to first disambiguate whether the content is new or repeated, and then evaluate its relevance to the specific objectives of their course.

Therefore, our work focuses primarily on the third category (i.e., similar content), as the effort required to determine whether the topic differs from the ones already included in previous responses can diminish the efficiency gains offered by the LLM. We address this challenge by minimizing human effort in determining whether overlapping topics in the new response are essentially equivalent, genuinely new, or too ambiguous to classify (see Figure 1). To achieve this, we adopt a more flexible definition of content similarity, allowing ambiguous material (i.e., at the center of the similarity spectrum) to be automatically labeled as mostly similar or mostly different. Simultaneously, this enables the filtering of the output produced by subsequent prompts to provide the instructor with the new content only. To achieve this, we rely on methods to maximize correct classification into the first two categories. Traditional lexical similarity techniques—such as cosine similarity or Jaccard index—perform well at the extremes, where content is either identical or entirely unrelated. However, they struggle in the middle of the spectrum, where nuanced semantic relationships must be understood and interpreted. More advanced approaches, including LLM-based methods, are also ill-suited for this task: they are costly, overly complex for simple topic categorization, and still inconsistent when dealing with borderline cases.

This involves applying a fuzzier definition of content similarity and thresholds that enable labeling ambiguous content as either mostly similar content or mostly different. By doing this, they can automatically be discarded or incorporated when compiling a new version of the outline. To this end, we use automated methods to maximize the content correctly classified in the first two categories. For instance, the novelty metric can be calculated using techniques that require minimal computational resources (i.e., Jaccard) and enable checking for complete similarity between the new content (i.e., list of topics in the outline generated in the latest iteration) and the existing content (i.e., list of topics in the initial or previous responses). Lexical similarity techniques, which perform very well at the extremes of the spectrum (i.e., where the content is either exactly the same or completely different), can be combined with other algorithms involving semantic similarity (e.g., transformer-based) to categorize the remaining content as mostly similar or mostly different, while maintaining low computational costs. Finally, the novelty metric utilizes two thresholds (i.e., t_{diff} and t_{same}) that define a range where new content that does not fit into either category requires additional review. As a result, NOV involves solving a two-

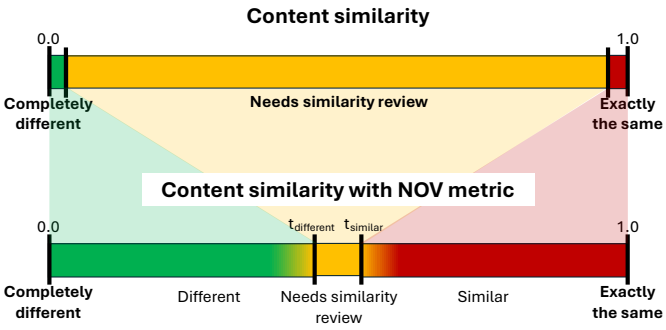

Fig. 1. Content similarity spectrum before and after content processing using the NOV metric.

fold optimization problem:

- maximizing F1 score in labeling the content as "different" or "same".
- reducing the distance between t_{diff} and t_{same} to avoid flagging too much content as requiring"similar", that is, requiring additional review effort.

Therefore, NOV aims to automatically categorize new content as novel (i.e., different), already seen (i.e., same), or to be reviewed (i.e., similar). This approach can be convenient for creating course outlines. For instance, the system can automatically run several prompt iterations, generate a structured course outline with a comprehensive list of topics, and provide the instructor with a final draft that removes repeated content. Nonetheless, in educational contexts, it can be utilized to draft syllabi, lecture plans, and other educational material.

IV. Study - Course outline generation

In our study, we applied the proposed methodology to a real-world scenario to demonstrate its feasibility, assess its performance, illustrate a use case for educational content creation, and discuss its benefits and implications based on empirical evidence.

A. Materials and Methods

We considered seven LLMs: ChatGPT o3, Claude 4 Sonnet, Gemini 2.5 Pro, DeepSeek R1, Grok 3, Qwen 2.5, and Llama 3.3. We used these models because they are among the most popular and widely used LLMs by individuals and institutions. We decided not to evaluate other commercially available systems, such as Copilot or Perplexity, because they are based on and utilize models like GPT (and others already included in our sample). Each LLM sampled for the study offers a unique perspective and reasoning style, including thinking models with Chain of Thought (CoT) reasoning. Although we focused on these seven models only, our methodology can be used with any LLM.

For our study, we focused on generating an outline for a course involving Computer Science knowledge, specifically Java, as it is one of the most widely taught and used programming languages. We then asked the LLMs to generate an outline for a 16-week course based solely on the model's internal knowledge (i.e., without supplying additional documents or information). We chose Java because it is an established language, commonly included in university curricula, and still relevant in industry-level applications. As a result, it is particularly suitable to be used as a standardized benchmark for evaluating how different LLMs generate structured educational content.

B. Protocol

Our data collection involved five people, each generating output from each of the LLMs considered. The LLMs were initially given a prompt asking them to create the course outline. The initial prompt provided detailed instructions on the course context, target audience, and output format. Concerning the latter component of the prompt, the LLM was instructed to organize the content into two levels of headings and a list of at least three topics for each subheading. Then, a different prompt (i.e., iteration prompt) asked the LLM to improve upon the previous response by revising the outline to expand it and add relevant topics while maintaining the same output format. The same prompt was utilized in each iteration, where the LLM was asked to further enhance its response by providing a revised list of topics a total of four times. The prompts used in our experiments are included in Appendix A.

At the end of the data collection process, we obtained 35 conversations, each consisting of five iterations. Although this number might seem small, each conversation involved a total of at least 300 topics (68 topics per iteration, on average), which resulted in a cumulative total of more than 23,700 individual course topics to be analyzed. The LLM output from every iteration was stored in markdown format in separate files, which were then parsed and pre-processed to remove non-relevant components of the response (e.g., introductory paragraphs and conclusions), retaining only the course outline.

Each conversation was analyzed separately, and the output of each iteration was compared only with the previous one. Pairwise similarity was calculated at the topic level using each of the string comparison techniques discussed earlier, resulting in a similarity score for each topic pair. Subsequently, we utilized the similarity score to calculate novelty metrics representing each iteration and the entire conversation. Finally, conversation-level metrics were aggregated by model to compare the strategies and performances of each LLM.

V. Results

After processing the markdown documents containing the lists of topics generated by the LLMs in the individual iterations of each conversation, we calculated simple statistics to describe content that was found to be exactly the same, completely different, and similar. The goal of this initial analysis was to evaluate topic similarity. As shown in Table I, out of 23,751 topics produced across five iterations, 10,278 (43.27%) topics resulted in different content, 4,821 (20.3%) were exactly the same, and 8,652 (36.43%) were flagged as requiring further examination by a human. The data show

that, in general, as the user iterates over the same response, LLMs continue to expand the outline, producing 30-60% of different topics, which highlights the need for an automatic disambiguation system.

TABLE I
ITERATION-WISE SIMILARITY DISTRIBUTION

	Output 2	Output 3	Output 4	Output 5	Total
Different	3089	2547	2150	2492	10278
Similar	2160	2332	2089	2071	8652
Same	724	1156	1517	1424	4821
Total	5973	6035	5756	5987	23751

Subsequently, we aggregated the data by model. Figure 2 represents the similarity distribution of the content generated by each model across the five iterations. As shown in the figure, Grok 3 achieved the highest percentage of different topics, followed by DeepSeek R1 and ChatGPT o3. On the contrary, Qwen 2.5 and Llama 3.3 demonstrated high redundancy and poor ability to produce new content through iterative prompting. Our findings suggest that, based on similarity metrics alone, Grok 3 outperforms the other models and should be preferred for creating course outlines. Nevertheless, the performances of DeepSeek R1 and ChatGPT o3 render them valid alternatives. However, even considering the three best-performing models, similarity metrics show that across the five iterations, Grok, DeepSeek, and ChatGPT respectively produced 43.55%, 43.51%, and 44.48% of content labeled as similar, which required further human revision. Items requiring human review contain either some degree of overlap with topics listed in previous iterations or a paraphrased version of a previously listed topic. For example, in one iteration, ChatGPT produced the topic *"Sorting & searching with Comparable and Comparator"*, while in the next, it generated *"Comparator, Comparable, and custom sorting; unmodifiable views"*. In cases like the one shown in the example, where words partially intersect, similarity algorithms such as Jaccard fail to accurately disambiguate the content as referring to the same topic or to two different ones. Therefore, these cases would be flagged for human review.

Instead, to automatically disambiguate content similarity, we computed the NOV score as described in the previous section. Specifically, we set thresholds t_{diff} and t_{same} at 0.46 and 0.54, respectively, because they were found to produce the most accurate results (see discussion below). Specifically, the NOV score was calculated using the Jaccard index to address topics with complete overlap, and then, by applying a second-stage filter that aggregated semantic measures (i.e., cosine similarity and transformer-based models). The NOV score allowed the categorization of the topics that were initially considered "similar" into either mostly different, mostly similar, or similar, which is shown in Figure 3.

Across all models and iterations, 36.43% of topics were initially considered similar (i.e., 8,652 of the total 23,751 topics), and thus, required human intervention. The NOV score

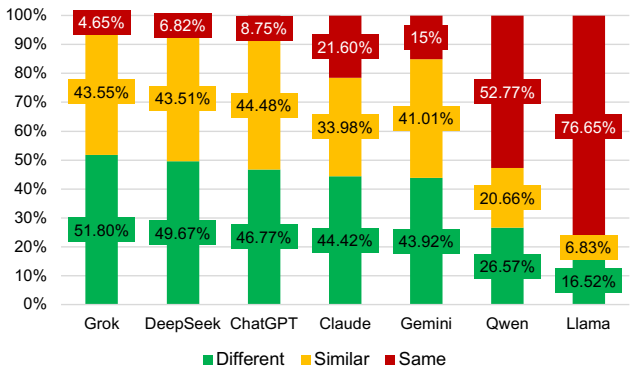

Fig. 2. Content similarity distribution by model across five iterations. Basic similarity metrics fail to categorize large portions of the content as either "different" or "same", resulting in additional manual review and effort for the instructor.

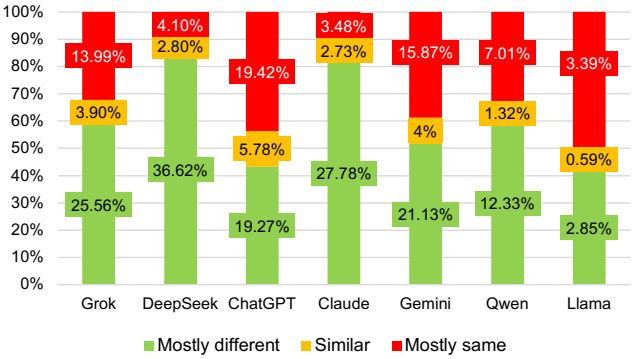

Fig. 3. Results of the NOV metric applied to the content flagged as "similar" by traditional similarity scores (see Figure 2). Data labels highlight how the NOV score automatically disambiguates most similar content into either mostly different (i.e., new) or mostly same (i.e., overlap).

enabled the automatic classification of the majority of the topics (i.e., 7,868 items) as either "mostly different" or "mostly similar". Consequently, as shown in Figure 4, only 3.3% of the topics were flagged as requiring manual review, saving a significant amount of human effort.

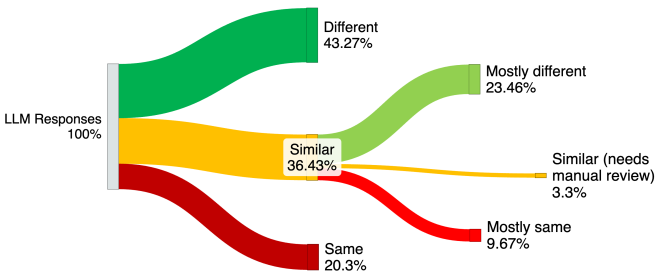

Fig. 4. Aggregated content similarity overview showing the similarity scores (shown in Figure 2) integrated with the result of the NOV score from Figure 3. NOV significantly reduces the number of "similar" items that require human review.

In addition to reducing human revision and improving the

overall iterative prompting workflow, the NOV score also provides a benchmark of the performance of the models in the context of iterative prompting. In Figure 5, to illustrate the overall new content produced by each model, we collapsed the four labels into two aggregate categories: *different* combines "different" and "mostly different," and *same* combines "same" and "mostly same". DeepSeek stood out as the best-performing model; 86.28% of its outputs across multiple iterations introduced new content. By comparison, Grok and Claude achieved 77.36% and 72.2%, respectively. ChatGPT and Gemini followed with 66.04% and 65.05%, while Qwen and Llama lagged significantly behind at 38.9% and 19.37%.

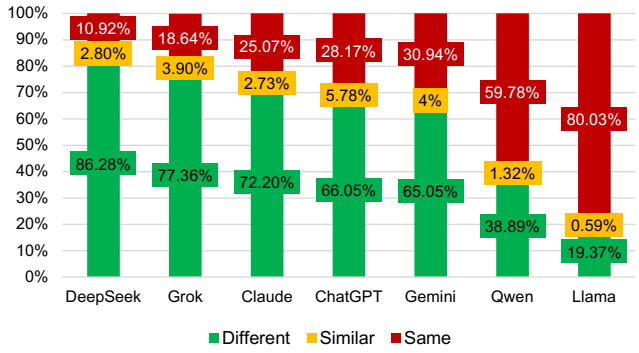

Fig. 5. NOV score by model (data labels highlight conversion of similar content into mostly different and mostly same content).

A. Performance Evaluation

Furthermore, we evaluated the performance of our method in terms of computational cost and accuracy to assess which algorithm obtains the best trade-off and whether our proposed approach can be efficiently applied to course outline generation.

The computational cost of similarity calculation using Jaccard, cosine similarity, and BERT was, on average, 0.0026 ms, 0.0187 ms, 0.203 ms, respectively, showing that even though transformer-based models required one order of magnitude more time than basic similarity metrics, they could be utilized efficiently for novelty calculation. In fact, combining the three methods resulted in an average computation time of 0.0745 ms. This is lower than BERT because the Jaccard index and Cosine similarity optimize calculations by filtering out topics that are exactly the same without requiring the use of BERT.

On the contrary, the LLM took an average of 1337.16 ms to complete one calculation. These results are also reported in Table II. Although all performances largely depend on the runtime environment, the difference in time required by the three algorithms and the LLM makes the latter solution the least suitable for our methodology, especially considering the performance of the model in terms of F1 score.

Each topic required an average of 36.86 comparisons, and each iteration involved an average of 128.65 topic pairs comparisons, for a total of approximately 5,000 calculations per iteration, on average, leading to an average additional computational cost of 70 ms, 220 ms, and 2,950 ms using Jaccard, Cosine similarity, and BERT, respectively. While the delay introduced by the first two algorithms is negligible, the longer processing times required by transformer-based models can be noticeable to the end-user and may affect their experience. However, the additional delay (i.e., ∼4 seconds on average) is still comparable with the time required by LLMs that refine their output through Chain of Thought (CoT), including thinking models.

Finally, we evaluated the accuracy of our approach in calculating the NOV score. To this end, we randomly selected a sample conversation for each model, resulting in a total of 35 iterations across seven models, involving 1090 topics (21% of the total). Then, we manually labeled each topic pair in each iteration as *mostly new* (i.e., 1.0) or *mostly repeated* (i.e., 0.0) content. Then, to evaluate the effectiveness of our approach, we calculated the accuracy metrics for the components of NOV (i.e., lexical, semantic, and LLM-based similarity analysis) and compared the alternative options. Our findings, reported in Table II, indicate that no single model configuration dominates in terms of quality metrics, necessitating trade-offs, which confirms the need for a scaffolded approach. Specifically, when used individually, Jaccard achieves high lexical precision and the best accuracy, but at the expense of very low recall, leaving many items unlabeled or incorrectly classified as negative. As a result, it works best for comparing content at the very end of the similarity spectrum. Compared to cosine similarity, transformer-based models capture deeper semantics without increasing false positives, making them ideal as a second-stage filter.

Our analysis also shows that using lexical and semantic pairing as a first-stage similarity filter helps recall without sacrificing much accuracy, minimizing the percentage of topics left uncategorized (i.e., 0.56%). Conversely, the LLM classifies almost everything as positive, resulting in the lowest precision and confirming that, even from an accuracy standpoint, it is unsuitable as a first pass. Although adding the LLM as the last-stage filter yields the strongest overall metrics with a very moderate skip, this comes at the cost of an additional trade-off regarding computational costs.

TABLE II
PERFORMANCE ANALYSIS OF INDIVIDUAL SIMILARITY METHODS (JCB INDICATES THE INTEGRATION OF JACCARD, COSINE, AND BERT)

Metric	Jaccard	Cosine	BERT	LLM	JCB
Compute time (ms)	0.0026	0.0187	0.203	1337.16	0.0745
Accuracy	73.98%	79.44%	79.73%	53.60%	79.27%
Precision	87.29%	78.16%	77.42%	50.81%	83.79%
Recall	49.24%	77.73%	80.23%	99.63%	67.88%
F1 score	62.96%	77.95%	78.80%	67.3%	75%
Skipped topics	6.81%	9.58%	5.43%	1.44%	9.76%

VI. Discussion

Current benchmarks primarily focus on aspects such as factual correctness or writing style, and they are not suitable for measuring whether LLMs genuinely introduce new concepts or merely rearrange existing ideas, an important distinction particularly for knowledge-driven contexts, including education. Instead, the proposed NOV score introduces a new metric for evaluating the output of LLMs in terms of new content added by each iteration. Indeed, novelty does not necessarily translate into relevance. Therefore, further review is required to determine whether the content flagged as "new" is worthy of consideration. Therefore, in the case of course outline generation, the instructor is ultimately responsible for determining whether a new topic included in a revised outline is appropriate in relation to the specific objectives of their course. Similarly, decisions regarding coherence with other course units, alignment with pedagogical standards, and progression across cognitive levels remain squarely within the instructor's domain. Nevertheless, by automatically disambiguating content, our framework provides instructors with a triage tool that filters out repeated information, helping them optimize their time and focus their attention on evaluating the appropriateness, coherence, and relevance of the new topics only. As a result, the objective of our approach is not to replace instructor judgment but to make their judgment easier and more targeted. This keeps pedagogical expertise at the center while reducing the effort spent sifting through repetitive model outputs.

Furthermore, the NOV metric could be incorporated into a more structured processing pipeline that evaluates relevance. To this end, after computing NOV scores, clustering algorithms (e.g., k-means or hierarchical clustering) could be applied to the topics requiring expert review. Clusters could then be ranked against course goals or seed exemplars using similarity scores, yielding shortlists of "relevant and novel" topics for instructor review. This would add two benefits: on the one hand, it would curate the long tail of novel items into coherent thematic sets. On the other hand, it would create recommendations based on the stated learning outcomes. As a result, rather than inspecting hundreds of isolated suggestions, instructors would receive a small number of labeled clusters with representative topics and rationales they can accept, revise, or reject.

More generally, the proposed NOV metric offers several advantages beyond applications in the field of education. NOV can be used in the context of iterative prompting to:

1) Automatically assign a score to the output of every iteration and use it to compile an aggregated outline, thus minimizing the effort required for human review.
2) Integrate the outputs of multiple conversations across different LLMs to draw from various sources.
3) Evaluate the performance of LLMs and benchmark their strategies and effectiveness in iterative prompting, aiding users in choosing a model.
4) Progressively and autonomously expand an outline vertically (i.e., specialize the topics) and horizontally (i.e., find additional topics) through automatic prompt iteration.
5) Enhance LLMs with an additional Chain of Thought (CoT) layer designed to maximize the performance of iterative prompting with minimal computational overhead.

Finally, the NOV score provides a practical means to benchmark LLMs, whether for instructional purposes or other types of applications. New LLMs are being released and updated frequently, and this trend is expected to continue in the future. Users, including instructors, will continually need to assess which models are most effective for their goals. NOV offers a straightforward method for generating tables of contents, structured outlines, and bullet-point lists from unstructured content with minimal repetition. Such benchmarking can guide model selection, inform procurement and policy, and help track improvements as models evolve.

VII. Conclusion and Future Work

We developed NOV, a systematic approach for measuring the novelty of content generated by LLMs through iterative prompting. In this paper, we focused on discussing the use of NOV in educational contexts, particularly for generating course outlines. Our evaluation across multiple models demonstrated that the proposed NOV score effectively distinguished genuinely new material from repeated or minimally altered outputs, thereby reducing the manual effort required to filter large volumes of content generated across multiple iterations.

Furthermore, we presented the results of a study in which we tested NOV with seven popular LLMs. Among the models, DeepSeek R1 produced the highest proportion of novel outputs across iterations. Indeed, these findings are specific to iterative prompting scenarios and should not be taken as a comprehensive ranking of model capabilities across all tasks. We also observed that models employing advanced reasoning strategies, such as CoT prompting, achieved stronger performance. Our results highlight the value of novelty detection as a decision-support tool for educators.

Building on the current work, we are also developing a system to simplify the evaluation of the relevance of content generated by LLMs by clustering topics by similarity. By doing this, alongside novelty, the framework could help users select the new topics generated at each iteration that most closely align with their goals. For instance, this addition could benefit instructors by clustering and ranking the new topics based on their relevance to the instructional goals of their course. Furthermore, we plan to automate iterative prompting, expand evaluations to a broader range of subject areas, and explore applications in generating assessments and adaptive learning pathways. Beyond education, potential use cases include technical documentation, legal drafting, and customized instructional manuals for industry.

References

[1] M. Giannakos, R. Azevedo, P. Brusilovsky, M. Cukurova, Y. Dimitriadis, D. Hernandez-Leo, S. Järvelä, M. Mavrikis, and B. Rienties, "The

promise and challenges of generative AI in education," *Behaviour & Information Technology*, vol. 44, no. 11, pp. 2518–2544, 2025.
[2] P. Sridhar, A. Doyle, A. Agarwal, C. Bogart, J. Savelka, and M. Sakr, "Harnessing LLMs in curricular design: Using GPT-4 to support authoring of learning objectives," *arXiv preprint arXiv:2306.17459*, 2023.
[3] A. K. Law, J. So, C. T. Lui, Y. F. Choi, K. H. Cheung, K. Kei-ching Hung, and C. A. Graham, "AI versus human-generated multiple-choice questions for medical education: a cohort study in a high-stakes examination," *BMC Medical Education*, vol. 25, no. 1, p. 208, 2025.
[4] Y. Zheng, X. Li, Y. Huang, Q. Liang, T. Guo, M. Hou, B. Gao, M. Tian, Z. Liu, and W. Luo, "Automatic Lesson Plan Generation via Large Language Models with Self-critique Prompting," in *International Conference on Artificial Intelligence in Education*. Springer, 2024, pp. 163–178.
[5] L. Yan, L. Sha, L. Zhao, Y. Li, R. Martinez-Maldonado, G. Chen, X. Li, Y. Jin, and D. Gašević, "Practical and ethical challenges of large language models in education: A systematic scoping review," *British Journal of Educational Technology*, vol. 55, no. 1, pp. 90–112, 2024.
[6] R. W. Tyler, "Basic principles of curriculum and instruction," in *Curriculum studies reader E2*. Routledge, 2013, pp. 60–68.
[7] H. Taba, "Curriculum development: Theory and practice," *Harcurt Brace*, 1962.
[8] G. P. Wiggins and J. McTighe, *Understanding by design*. Ascd, 2005.
[9] R. Sajja, Y. Sermet, D. Cwiertny, and I. Demir, "Platform-independent and curriculum-oriented intelligent assistant for higher education," *International Journal of Educational Technology in Higher Education*, vol. 20, p. 42, 2023.
[10] H. Fan, G. Chen, X. Wang, and Z. Peng, "LessonPlanner: Assisting novice teachers to prepare pedagogy-driven lesson plans with large language models," in *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, 2024, pp. 1–20.
[11] A. Attard and A. Dingli, "Empowering educators: Leveraging large language models to streamline content creation in education," *International Journal of Information and Education Technology*, vol. 15, no. 9, pp. 2021–2030, 2025.
[12] I. Solaiman, T. Maria, and M. Milanova, "Educational content generation using multi-LLM agents," *International Robotics & Automation Journal*, vol. 10, no. 3, pp. 85–87, 2024.
[13] B. Hu, J. Zhu, Y. Pei, and X. Gu, "Exploring the potential of LLM to enhance teaching plans through teaching simulation," *npj Science of Learning*, vol. 10, no. 1, p. 7, 2025.
[14] A. Dornburg and K. J. Davin, "ChatGPT in foreign language lesson plan creation: Trends, variability, and historical biases," *ReCALL*, pp. 1–16, 2024.
[15] Q. Bao, J. Leinonen, A. Y. Peng, W. Zhong, G. Gendron, T. Pistotti, A. Huang, P. Denny, M. Witbrock, and J. Liu, "Exploring iterative enhancement for improving learnersourced multiple-choice question explanations with large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 28, 2025, pp. 28 955–28 963.
[16] V. Ashrafimoghari, N. Gürkan, and J. W. Suchow, "Evaluating large language models on the GMAT: Implications for the future of business education," *arXiv preprint arXiv:2401.02985*, 2024.
[17] A. Pack, A. Barrett, and J. Escalante, "Large language models and automated essay scoring of English language learner writing: Insights into validity and reliability," *Computers and Education: Artificial Intelligence*, vol. 6, p. 100234, 2024.
[18] V. Ghimire, J. Shrestha Lama, B. Dhungana, N. Sadat, and N. Caporusso, "Evaluating Output Novelty in Iterative Prompting in Educational Content Generation," to appear in *Proceedings of the 17th International Conference on Education Technology and Computers (ICETC)*, 2025.

APPENDIX A
LLM PROMPTS USED IN OUR EXPERIMENTS

A. Initial Prompt

Context: You are tasked with designing a 16-week course outline for a college-level course on Object-Oriented Programming (OOP) using Java. The course should provide a structured and progressive learning path, covering foundational concepts, advanced techniques, and real-world applications. The outline should include at least three levels of headings, ensuring depth and clarity. Each week should build upon the previous, culminating in a final project that integrates key concepts.

Role: You are an expert computer science professor with over two decades of experience teaching Java and OOP principles at the university level. You have deep knowledge of software design patterns, best practices, and industry applications. Your teaching philosophy emphasizes hands-on learning, problem-solving, and real-world project development.

Action:

1) Structure the course into 16 weeks, ensuring a logical progression from basic to advanced topics.
2) Use at least three levels of headings to organize the outline into clear sections (e.g., Weeks $\rightarrow$ Topics $\rightarrow$ Subtopics).
3) Include fundamental Java programming concepts before moving into advanced OOP topics like design patterns and frameworks.
4) Integrate weekly hands-on assignments, quizzes, or exercises to reinforce learning.
5) Allocate time for code optimization, debugging techniques, and best coding practices.
6) Include at least one major project in the later weeks, requiring students to apply multiple concepts learned.
7) Ensure the outline prepares students for real-world applications.

Format: The course outline should be structured using hierarchical headings (e.g., H1 $\rightarrow$ H2 $\rightarrow$ H3) to ensure clarity. Each week should be clearly labeled, and subtopics should be included to break down the material further.

Target Audience: The course is designed for undergraduate computer science students or software engineering majors who have basic programming knowledge but are new to object-oriented programming and Java. The material should be structured to cater to both students with prior experience in another language and those new to OOP concepts.

B. Follow-Up Prompt

Could you improve the course outline further by adding anything that is missing (making it more comprehensive), could add value to the course, relevant to the real world, and could help enhance students' understanding or learning?