

Mind the Gap: An Educator-Oriented Hybrid AI Platform for Iterative Review of Course Material

Valerio Crocetti^{1,2}, Elia Pacioni^{1,3} and Davide Calvaresi¹

¹HES-SO Valais-Wallis, 3960 Sierre, Switzerland

²Università Politecnica delle Marche, 60131, Ancona, Italy

³University of Extremadura, 06800 Mérida, Spain

Abstract

Generative AI (GenAI) can complete assessed tasks without students demonstrating the intended skills. Educators must therefore review and update course material, including syllabi, briefs, and rubrics, to address task trivialization. Such a review entails assessing course material against custom pedagogical criteria, including constructive alignment, rubric quality, accessibility, and GenAI-driven task trivialization. This paper presents Mind the Gap (MTG), an educator-oriented proof-of-concept platform for interactive review of course material. The platform supports iterative redesign through document versioning and comparison of review snapshots. Plain-English requirements are compiled into Common Expression Language (CEL) rules with extraction contracts. Language models report observations for each page, while deterministic code aggregates evidence and evaluates compliance. Two dedicated agents support document retrieval and interactive refinement of redesign suggestions. MTG can run on-premises on machines such as NVIDIA DGX Spark, keeping document processing and storage on institution-controlled infrastructure. Exploratory reviews of institutional, open-access, and deliberately flawed course material produced findings about objectives, unstated requirements, GenAI exposure, and surface defects. In these runs, documents considered well-formed raised no findings, and correcting flagged defects removed the corresponding findings. Further evaluation, including error rates and usefulness as assessed by educators reviewing their own courses, is planned for future work.

Keywords

course design, educator-facing AI, hybrid AI, rule compilation, auditable AI, local deployment, higher education

1. Introduction

Essays, laboratory reports, problem sets, and take-home exams provide evidence of what a student can do. When generative AI (GenAI) supplies the requested answer, students can bypass the reasoning or practice the task was designed to elicit. Institutional guidance addresses this problem unevenly. A study found that just under half of the world's fifty top-ranked institutions published advice on GenAI covering academic integrity, communication with students, and assessment design [1]. To categorize such interventions, Corbin et al. distinguish between a *discursive* change, instructing students while leaving the task as it was, and a *structural* change, altering the mechanics of the task itself [2].

Assessment redesign also affects the curriculum. Educators must reconsider what students are expected to demonstrate and keep revised learning outcomes consistent across course documents. Recent guidance points toward process evidence, oral verification, and authentic judgment [3, 4]. Individual instructors often decide which measures to adopt. Even institutions with strict rules usually let instructors adjust their course-level GenAI policies [5].

This redesign spans syllabi, slides, assignments, and grading rubrics. Pedagogical frameworks can guide the review of this course material by making explicit how learning outcomes, teaching activities, and assessment should relate and what cognitive processes they require. Constructive alignment addresses these relationships by linking what students are expected to learn with what they practice and demonstrate in assessment [6]. Rubrics should specify criteria for judging the

AI-LEAP 2026 - 1st International Workshop on AI Learning, Education And Practice, co-located with AIXIA 2026, University of Perugia, October 2026

✉ valerio.crocetti@hevs.ch (V. Crocetti); elia.pacioni@hevs.ch (E. Pacioni); davide.calvaresi@hevs.ch (D. Calvaresi)

🆔 0009-0002-3760-6041 (V. Crocetti); 0000-0002-1557-4870 (E. Pacioni); 0000-0001-9816-7439 (D. Calvaresi)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

expected performance. For example, if an outcome requires source evaluation, teaching activities should give students practice evaluating sources, and the assessment should ask them to justify their judgments. Bloom’s revised taxonomy helps specify the cognitive demands of these outcomes and tasks by distinguishing remembering, understanding, applying, analyzing, evaluating, and creating [7]. These categories help educators check whether an assessment requires the type of thinking stated in an outcome. A performance-based objective expresses that outcome as an observable action, such as comparing two sources and justifying which is more credible. For tasks that GenAI can complete, this review must also examine how the assessment elicits evidence of the student’s own performance.

Large language models (LLMs) can assist with the review of course material. Asking a model to judge a document against a pedagogical requirement combines interpretation, evidence selection, and the decision in a single step. In a study of student-work grading, an LLM did not reliably follow its rubric-based evaluation instructions [8]. Judging a document against a stated criterion presents a similar challenge, and models are steadier on explicit textual features than on inferred properties [9]. An incorrect judgment can therefore be difficult to trace to its source.

Course adaptation requires repeated reviews as course material and pedagogical requirements change. A revised learning objective may require changes to the assignments and rubrics that assess it, while the criteria used in the review depend on the course and its teaching context. Mind the Gap (MTG) supports this process through a workspace where educators define requirements, inspect findings against the source pages, and review revised course material. This paper contributes a working prototype combining requirements compiled into executable rules, educator approval of proposed rules, and comparison across review runs. The platform can be deployed on institution-controlled infrastructure. Evaluation with educators reviewing their own courses remains the central open question.

Section 2 discusses related work. Section 3 presents the course workspace and the rule mechanism. Section 4 describes the implementation, and Section 5 reports exploratory observations. Section 6 discusses limitations and next steps, followed by the conclusion in Section 7.

2. Related Work

GenAI applications in higher education address both students’ learning activities and educators’ work in course preparation and assessment. Learner-facing systems offer tutoring [10], help with course-work [11], and guidance for self-regulated learning [12]. Educator-facing systems support the design of these activities and the assessment of student work. Instructional Agents generates course content [13], Feedback Copilot supports feedback on assignments [14], and GradeHITL involves educators in refining rubrics for automated grading [15]. Deployment frameworks also give educators control over the sources and infrastructure used by these services. LAMB grounds conversational agents in curated sources [16], and Reza et al. provide a local small-model stack for exercises, rubrics, and grading [17].

MTG shares the emphasis on educator control and institutional deployment, applying it to the review of existing course material against explicit pedagogical requirements. Its contribution combines executable review criteria with evidence inspection and comparison after revisions. The underlying mechanism builds on AiDER [18], a document-review pipeline that separates model-reported evidence from deterministic rule evaluation. AiDER compiles natural-language requirements into Common Expression Language (CEL) [19] conditions and derives the evidence required for each document check. MTG retains this separation and specifies evidence through named document fields and explicit extraction contracts. Within MTG, educators approve proposed rules, inspect findings with supporting evidence and redesign suggestions, then revise their material and compare results across reviews.

3. System Overview

The prototype is a web application organized around the *course workspace*, which holds a course’s documents, review rules, and previous reports. An educator uploads course material, assigns document types (syllabus, slides, assessment, rubric), and selects rules from a default catalog. Each *rule* checks a

pedagogical *requirement*, such as asking a syllabus to state an observable learning objective. Educators can also write custom requirements in plain English, which MTG compiles into proposed rules for their approval. A review applies the selected rules and returns *verdicts*. Unmet requirements appear as findings with supporting evidence and redesign suggestions. Figure 1 shows this sequence and the educator’s decisions.

The architecture follows six design goals. Grounded analysis (G1) binds every finding to the uploaded document, version, and source page. Explicit criteria (G2) make each check inspectable through a rule scoped by document type and activated per course. Separating findings from advice (G3) preserves the computed verdict when educators edit generated redesign suggestions. Human-in-the-loop review (G4) gives educators control over document classification, requirements, generated rules, and revisions. Iterative redesign (G5) uses immutable versions and frozen snapshots to compare later reviews with earlier ones. Institutional deployment (G6) keeps processing and storage on institution-controlled infrastructure.

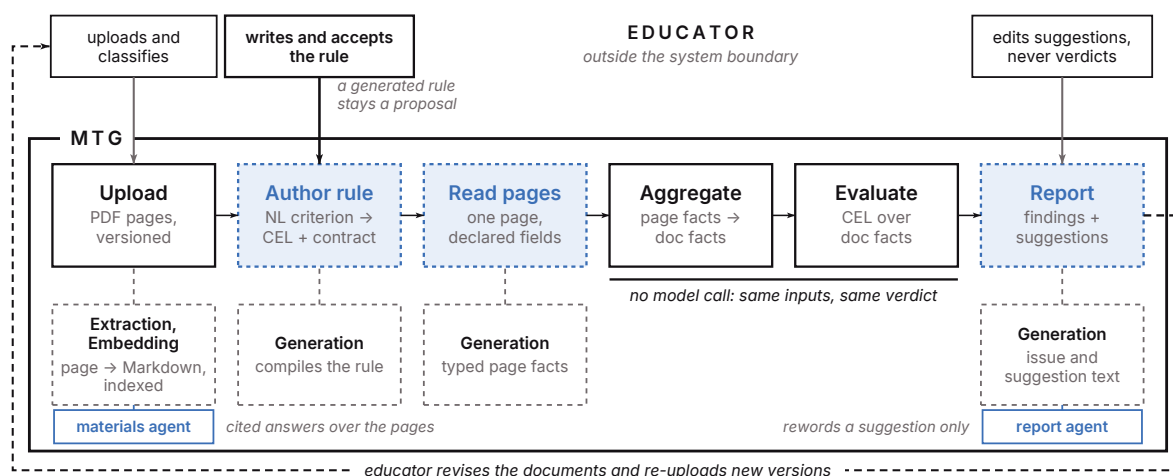


Figure 1: A course review proceeds in sequence. The educator classifies documents, accepts or rejects rules, and edits suggestions. Rule authoring precedes page reading, as a rule’s extraction contract instructs the model on which facts to observe for each page. The return path at the bottom closes the loop outside the system.

Document intelligence stores each upload as a document version and renders its pages as images. A document-parsing model transcribes them into markdown, preserving page boundaries and layout. The transcribed pages are indexed for retrieval within the course. Unchanged pages reuse their transcription and matching extracted observations across versions, even if their position changes.

Hybrid analysis applies the active rules to processed documents (Section 3.2). *Reporting* groups the results by document and records each check as passed, failed, not applicable, or failed to evaluate. For each failed check, the model writes an issue description and a redesign suggestion.

3.1. The Course Workspace

Figure 2 follows successive reviews of course material in four steps: uploading and classifying documents, choosing review criteria, inspecting findings and suggestions, and comparing results after a revision.

MTG provides a default catalog covering learning outcomes, assessment design, AI-readiness, accessibility, and rubric quality. Its rules derive from the teaching-and-assessment literature and the project’s faculty focus groups. Educators select relevant rules and can propose additional requirements in plain English. What counts as a suitable assessment can differ between a first-year survey and a final-year project, and between colleagues teaching the same subject. Each rule displays its requirement and rationale upfront, with the CEL condition available for detailed inspection (G2). Custom proposals await the educator’s acceptance, revision, or rejection before use (G4).

After revising a syllabus or assignment, an educator can run the review again to examine the effect

1. UPLOAD AND CLASSIFY

2. DEFINE YOUR CRITERIA

3. VERDICTS WITH EVIDENCE

4. COMPARE RUNS

of those changes. Re-uploading a revised file creates a new version of the same document and preserves the earlier one. Each review stores a *snapshot*, a fixed record of its results. Stable rule codes identify the same checks across runs. Comparing two snapshots shows which findings were resolved, which remain, and which appeared after the revision (G5). Earlier reports retain the results recorded at the time of each review, allowing the educator to return to previous stages of the redesign.

Each verdict opens to show the evidence used in the check: the values extracted from the document, the supporting quotations, and their source pages (G1). Findings and suggestions appear in adjacent columns, with suggestions available for editing (G3). The report orders findings by severity and groups them by improvement theme. The AI-readiness score is the severity-weighted proportion of passed applicable checks in that category. Not-applicable and failed-to-evaluate checks are excluded. The educator can follow a finding back to the document, assess the evidence, and decide whether the suggested change is appropriate.

Two agents support further work with the material and the report. The *materials agent* answers questions about the uploaded documents through retrieval-augmented generation [20], citing filenames and pages. It can help locate a topic, clarify what an assignment asks, or compare statements in different documents.

The *report agent* helps educators refine suggestions or adapt them into document-ready prose. Educators give natural-language instructions and can target documents using @-mentions in the chat. The agent updates suggestion text while the report retains the computed verdicts and supporting evidence.

3.2. Rules, Contracts, and Verdicts

A rule specifies how a requirement is checked against a document. It contains the requirement, its pedagogical *rationale*, an *extraction contract*, and a *CEL condition*. The contract specifies the facts to extract from each page and how code aggregates their values for the document. The condition uses those combined values to determine whether the requirement is satisfied. Severity and category describe the importance of a failure and the improvement theme it belongs to.

For example, the default rule `sy1`, based on Biggs and Bloom [6, 7], requires a syllabus to state at least

one learning objective as an observable performance. Its contract requests a Boolean (true/false) value indicating whether each page states such an objective. The values are combined using the aggregator any, and the CEL condition passes the document when at least one page qualifies (Figure 3).

Each contract field declares a value type, an observation instruction, and an aggregation operator. The operator must reflect the requirement’s scope. any checks for one qualifying page, while all requires every applicable page to qualify. A requirement covering every objective within a page also needs an observation instruction that checks each objective.

An unsuitable operator can produce an executable rule that checks a weaker requirement, a problem we refer to as the *coverage trap*. In a fifty-page slide deck, for example, a single qualifying page would satisfy an any condition even if the educator intended a requirement to hold throughout the document. The operator is therefore included in the proposal for educator inspection. Contracts may also declare an *applicability gate*: for example, a requirement about permitted AI use applies when a document contains an assessment task. When this precondition is absent, the check is reported as not applicable.

During rule generation, validation checks that the CEL condition parses, that aggregation operators match field types, and that the condition and contract agree on the extracted fields. Invalid proposals enter a bounded repair loop [21]. The model receives the validation errors and attempts to correct the proposal. Unresolved errors leave it in a visible failed state. These checks establish structural validity. The educator’s review addresses whether the proposal expresses the intended requirement and suits the course.

A review applies each active rule to document versions matching its assigned document types. Rules that need the same facts share an extraction request. The model reads each page image with its transcription as support, returning the requested values, applicability flags, and quotations. Constrained decoding [22] enforces this output structure. Code then aggregates the page values and evaluates the CEL condition. The resulting verdict retains the supporting quotations and page references for inspection. The CEL condition reads the aggregated values, while the quotations provide the educator with the passages behind those values.

Processing errors, such as a page that could not be read, are reported separately and excluded from scoring.

These rules evaluate each document independently. To examine alignment across the course, MTG also asks the model to compare the syllabus with summaries of the other documents. This analysis considers whether the stated objectives appear in teaching material and are exercised by graded work. The report presents this model-generated assessment in a separate section from the rule verdicts.

4. Implementation

The prototype implements the complete review workflow, including both agents and comparison across runs. The default catalog contains twenty-two rules over four document types. Regression tests cover aggregation, applicability, and error handling.

Background workers handle document rendering, transcription, and model invocations. Additional workers allow concurrent evaluation of multiple courses. Model serving stays outside the application, in three roles: a document-parsing model transcribing rendered pages, a generation model that reads pages against contracts, compiles rules, and writes report prose, and an embedding model indexing the pages for retrieval. Each role connects through an OpenAI-compatible endpoint and can be served on separate hardware. Our models run on an NVIDIA DGX Spark workstation¹ and are served by vLLM [23], alongside the application and its workers, which run as versioned containers under a single Docker Compose file (G6).

For the generation role, we tested local models from 2B to 35B in dense and sparse mixture-of-experts (MoE) configurations [24, 25], and currently use a quantized MoE model in the 26–35B class among the open-weight models tested. Report generation drove model selection. Smaller models produced

¹<https://www.nvidia.com/en-us/products/workstations/dgx-spark/>

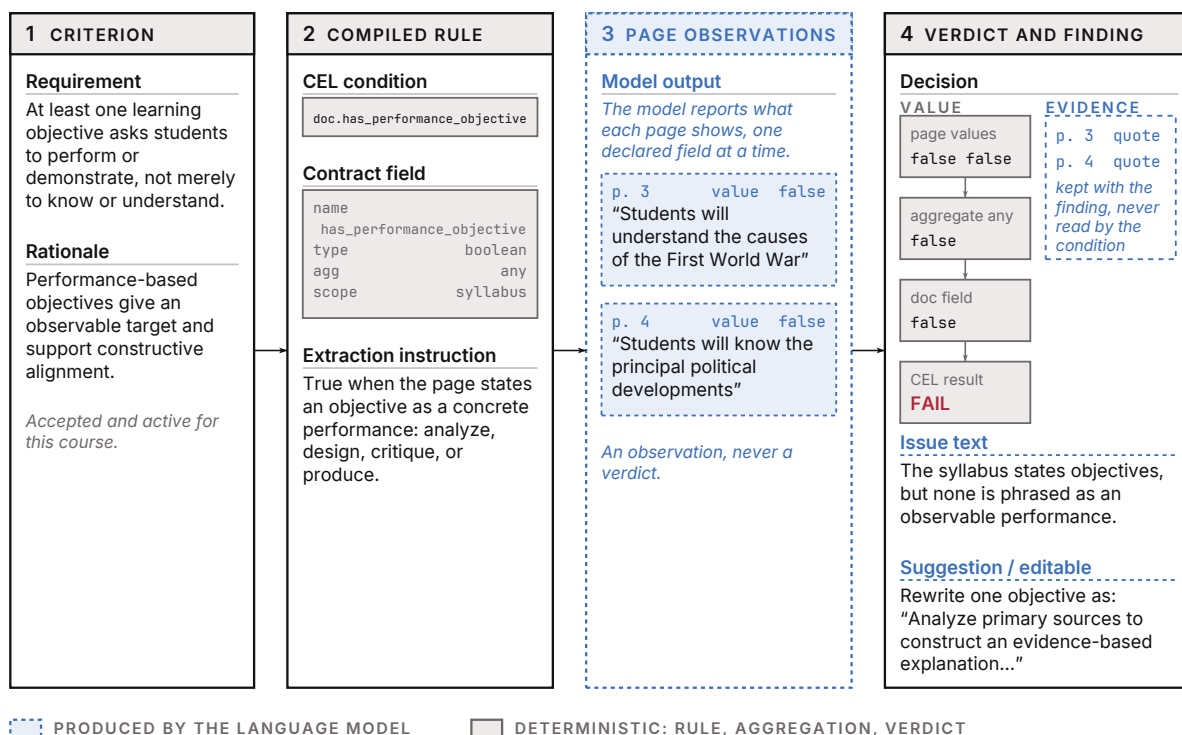


Figure 3: Example of a syllabus check. (1) The accepted rule includes the requirement and its rationale. (2) The contract defines the observation to collect and the any aggregation operator. (3) Both pages state objectives using “understand” or “know”, so both observations are false. (4) Code aggregates these values and evaluates the CEL condition, producing a failed verdict linked to the quotations and pages. The model then writes the issue and redesign suggestion.

incomplete reports and repeated wording across documents. One model covers page reading, rule authoring, reporting, and both agents and is therefore sized for the most demanding of them.

5. Exploratory Observations

We ran exploratory checks on material from courses taught at our institutions, the open-access course material used in AiDER [26], and documents assembled with intentional defects. We report qualitative observations without benchmark metrics or baseline comparisons. Four families of findings emerged from the default rule set. Alignment findings included objectives stated at recall level and rubric criteria that did not map to the outcomes a task claimed to assess. Specification findings included grading components named without weights and a missing accommodation policy. GenAI-exposure findings included missing policies on permitted tool use, final deliverables collected without process evidence, and prompts a model could answer unaided, such as an essay on “the main causes of the French Revolution”. Surface findings included misspellings and internally inconsistent assessment items, such as a question contradicting the data it supplied.

In these runs, documents considered well-formed raised no findings, including those containing dates, proper nouns, and domain terms. Correcting flagged defects removed the corresponding findings on the next run, while unchanged documents kept their verdicts. These observations show how findings changed after revision. Without systematically annotated cases, they cannot establish reliability or estimate a false-positive rate.

6. Discussion, Limitations, and Next Steps

MTG has not been studied with educators reviewing their own courses, so its usability, usefulness, and effects on redesign decisions remain untested. Its scope covers syllabi, assessments, rubrics, and uploaded teaching material. Whether classroom activities support the cognitive levels declared in a syllabus cannot be established from these documents.

The separation of observations, verdicts, and suggestions allows educators to inspect how a finding was produced and consider the proposed revision. Its practical value depends on whether they can interpret the evidence, identify unsuitable rules, and adapt the advice to their teaching context.

Transcription and extraction errors can produce incorrect verdicts even when the CEL condition executes correctly. The workspace retains page images, transcriptions, and quotations for inspection. Although code links findings to their source pages, the accuracy of the extracted values and quotations remains dependent on the model. Generated explanations and suggestions can misrepresent the evidence or propose revisions that do not suit the course, so they also require educator review. The current implementation does not verify that each quotation reproduces the page verbatim. Automated quotation checks remain to be implemented. The original page and its transcription allow educators to inspect disputed findings in context.

Four directions follow. First, educators should be able to test a proposed rule on the course's pages before accepting it. Second, smaller models produced incomplete reports, and suggestions still use generic course-design language. We intend to fine-tune on pedagogical literature and educator interactions to address these problems. Third, rule evaluation should expand from single-document contracts to a multi-document scope, letting deterministic CEL expressions operate over facts aggregated across the course material. Fourth, evaluation should measure errors against educator-annotated cases and compare baseline approaches, alongside a field study of how educators interpret findings and use suggestions.

7. Conclusion

Adapting a course to GenAI requires educators to decide what an assessment should measure and revise the documents that define and assess those outcomes. This work involves repeated review, course-specific criteria, and, where institutional policies require it, local processing. MTG gives educators a review they can configure, inspect, and repeat on institution-controlled infrastructure. Separating model-reported evidence from rule evaluation allows educators to challenge both the observations and the criteria behind a finding. We implemented the workspace and exercised its review and revision cycle on course material. The next step is to test whether educators reviewing their own courses recognize the problems MTG identifies and use its suggestions to revise their course material.

Acknowledgments

This work was supported by the HES-SO R&I project *Mind the Gap: Adapting Higher Education to the Age of AI* (MTG, 140012/RI-PSTRAT25-06).

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT and Claude for grammar, style, and clarity checks and sentence-level paraphrasing. Grammarly was also used to support grammar, spelling, and style checks. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

- [1] B. L. Moorhouse, M. A. Yeo, Y. Wan, Generative AI tools and assessment: Guidelines of the world's top-ranking universities, *Computers and Education Open* 5 (2023) 100151. doi:10.1016/j.caeo.2023.100151.
- [2] T. Corbin, P. Dawson, D. Liu, Talk is cheap: why structural assessment changes are needed for a time of GenAI, *Assessment & Evaluation in Higher Education* 50 (2025) 1087–1097. doi:10.1080/02602938.2025.2503964.
- [3] J. M. Lodge, S. Howard, M. Bearman, P. Dawson, et al., *Assessment Reform for the Age of Artificial Intelligence*, Technical Report, Tertiary Education Quality and Standards Agency, Melbourne, Australia, 2023. URL: <https://www.teqsa.gov.au/guides-resources/resources/corporate-publications/assessment-reform-age-artificial-intelligence>.
- [4] D. Nikolić, M. Basta Nikolić, Designing AI-resilient assessment in higher education: a four-pillar conceptual framework, *Frontiers in Artificial Intelligence* 9 (2026) 1841682. doi:10.3389/frai.2026.1841682.
- [5] P. Xiao, Y. Chen, W. Bao, Waiting, banning, and embracing: An empirical analysis of adapting policies for generative AI in higher education, *arXiv preprint arXiv:2305.18617* (2023).
- [6] J. Biggs, Enhancing teaching through constructive alignment, *Higher Education* 32 (1996) 347–364. doi:10.1007/BF00138871.
- [7] L. W. Anderson, D. R. Krathwohl (Eds.), *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*, Longman, New York, 2001.
- [8] C.-H. Chiang, W.-C. Chen, C.-Y. Kuan, C. Yang, H.-y. Lee, Large language model as an assignment evaluator: Insights, feedback, and challenges in a 1000+ student course, in: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Miami, FL, USA, 2024, pp. 2489–2513. doi:10.18653/v1/2024.emnlp-main.146.
- [9] D. Hauk, N. Soujon, How reliable are large language models in analyzing the quality of written lesson plans? A mixed-methods study from a teacher internship program, *Computers and Education: Artificial Intelligence* 10 (2026) 100538. doi:10.1016/j.caeai.2025.100538.
- [10] W. Xie, H. Shi, Y. Li, J. Liu, STAP: A Socratic Tutor for Adaptive Programming with Pedagogical Scaffolding, in: *Proceedings of the 2025 2nd International Symposium on Artificial Intelligence for Education, ISAIE '25*, Association for Computing Machinery, New York, NY, USA, 2025, pp. 584–587. URL: <https://doi.org/10.1145/3775073.3775165>. doi:10.1145/3775073.3775165.
- [11] J. Zamfirescu-Pereira, L. Qi, B. Hartmann, J. DeNero, N. Norouzi, 61A Bot Report: AI Assistants in CS1 Save Students Homework Time and Reduce Demands on Staff. (Now What?), in: *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1, SIGCSETS 2025*, Association for Computing Machinery, New York, NY, USA, 2025, pp. 1309–1315. URL: <https://doi.org/10.1145/3641554.3701864>. doi:10.1145/3641554.3701864.
- [12] X. Li, T. Li, L. Yan, Y. Li, L. Zhao, M. Raković, I. Molenaar, D. Gašević, Y. Fan, FLoRA: An advanced AI-powered engine to facilitate hybrid human-AI regulated learning, *Computers & Education* 243 (2026) 105527. URL: <https://www.sciencedirect.com/science/article/pii/S0360131525002969>. doi:10.1016/j.compedu.2025.105527.
- [13] H. Yao, W. Xu, J. Turnau, N. Kellam, H. Wei, Instructional agents: Reducing teaching faculty workload through multi-agent instructional design, in: *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2026, pp. 4087–4109. URL: <https://aclanthology.org/2026.eacl-long.191/>.
- [14] S. Pozdniakov, J. Brazil, S. Abdi, A. Bakharia, S. Sadiq, D. Gašević, P. Denny, H. Khosravi, Large language models meet user interfaces: The case of provisioning feedback, *Computers and Education: Artificial Intelligence* 7 (2024) 100289. doi:10.1016/j.caeai.2024.100289.
- [15] Y. Chu, H. Li, K. Yang, Y. Copur-Gencturk, J. Tang, LLM-based automated grading with human-in-the-loop, in: *2025 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, IEEE, Macao, 2025, pp. 1–8. doi:10.1109/TALE66047.2025.11346771.
- [16] M. Alier, J. Pereira, F. J. García-Peñalvo, M. J. Casañ, J. Cabré, LAMB: An open-source software

framework to create artificial intelligence assistants deployed and integrated into learning management systems, *Computer Standards & Interfaces* 92 (2025) 103940. doi:10.1016/j.csi.2024.103940.

- [17] Z. Reza, A. Mazur, M. T. Dugdale, R. Ray-Chaudhuri, Small models, big support: A local LLM framework for educator-centric content creation and assessment with RAG and CAG, *arXiv preprint arXiv:2506.05925* (2025). doi:10.48550/arXiv.2506.05925.
- [18] V. Crocetti, E. Pacioni, A. F. Dragoni, M. I. Schumacher, D. Calvaresi, AiDER: Auditing and document evaluation via rule compilation and small language models – an education case study, in: *AIxIA 2026 – Advances in Artificial Intelligence: 24th International Conference of the Italian Association for Artificial Intelligence*, Perugia, Italy, October 6–9, 2026, *Proceedings, Lecture Notes in Artificial Intelligence*, Springer Nature Switzerland, Cham, 2026. To appear.
- [19] Google, Common expression language (CEL), <https://cel.dev/overview/cel-overview>, 2023. Accessed 28 July 2026.
- [20] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive NLP tasks, in: *Advances in Neural Information Processing Systems*, volume 33, Curran Associates, Inc., 2020, pp. 9459–9474.
- [21] V. Crocetti, A. F. Dragoni, Bridging large language models and logic programming, in: *2025 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRINE)*, 2025, pp. 1066–1071. doi:10.1109/MetroXRINE66377.2025.11340308.
- [22] B. T. Willard, R. Louf, Efficient guided generation for large language models, 2023. doi:10.48550/arXiv.2307.09702. *arXiv:2307.09702*.
- [23] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, I. Stoica, Efficient memory management for large language model serving with PagedAttention, in: *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP '23*, Association for Computing Machinery, Koblenz, Germany, 2023, pp. 611–626. doi:10.1145/3600006.3613165.
- [24] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, J. Dean, Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, in: *5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017. URL: <https://openreview.net/forum?id=B1ckMDqlg>, conference track paper.
- [25] W. Fedus, B. Zoph, N. Shazeer, Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, *Journal of Machine Learning Research* 23 (2022) 1–39. URL: <http://jmlr.org/papers/v23/21-0998.html>.
- [26] K. McMurray, Business communications 2, Course pack, BCcampus Open Collection, 2020. URL: <https://collection.bccampus.ca/course-pack/cntcnGgB/>, curated and designed by Karen McMurray; course based on BENG 150; released through the BCcampus Open Online Courses Project in 2021; CC BY 4.0 except where otherwise noted; accessed 2026-05-27.