AiDER: Auditing and Document Evaluation via Rule Compilation and Small Language Models - An Education Case Study

Valerio Crocetti[1,2], Elia Pacioni[1,3], Aldo Franco Dragoni[2], Michael Ignaz Schumacher[1], and Davide Calvaresi[1]

[1] HES-SO Valais-Wallis, 3960 Sierre, Switzerland
{elia.pacioni, valerio.crocetti, michael.schumacher, davide.calvaresi}@hevs.ch, a.f.dragoni@univpm.it,
[2] Università Politecnica delle Marche, 60121, Ancona, Italy
[3] University of Extremadura, 06800 Mérida, Spain

Abstract. Reviewing documents against natural-language requirements is hard when those requirements demand interpretation, contextual evidence, and non-trivial judgment. It is even harder when the review is to be automated reliably and made auditable. In domains like education, where teaching materials must align with pedagogical models and evolving expectations, including the rise of Large Language Models (LLMs), such decisions cannot be reduced to mechanical checks. When an LLM assigns a verdict directly, interpretation, evidence selection, and judgment collapse into a single opaque step, making errors hard to localize and outcomes hard to audit.

AiDER separates interpretation from judgment under a deliberate 2B–4B parameter constraint. Each natural-language requirement is compiled into an executable Common Expression Language (CEL) rule, and static analysis of that rule produces a bounded-evidence scaffold that constrains what the model may extract from the document. The verdict is then computed deterministically by applying the rule to the extracted evidence (no LLM call at this step), yielding an inspectable trace from requirement to rule, rule to evidence, and evidence to verdict.

We evaluate AiDER on four small models (Gemma 4 and Qwen3.5, at 2B and 4B) under three compilation strategies: human-written reference rules, direct compilation, and validation-loop repair. The evaluation uses 200 requirement-document pairs from a higher-education course pack. Across three performance dimensions (syntactic validity, semantic fidelity, and document-level agreement), direct compilation often yields valid but semantically divergent rules, whereas validation-driven repair substantially improves both validity and fidelity across all four models. An ablation shows that the static scaffold alone encodes most of the extraction intent: LLM-generated extraction contracts add little when the rule is well-specified, but matter more when it is ambiguous.

Keywords: rule compilation · pedagogical review · small language models · neuro-symbolic AI · hybrid AI · auditable AI

1 Introduction

The rapid progress of Generative Artificial Intelligence (GenAI) has sparked interest in automating review tasks long regarded as matters of human judgment. In many domains, documents must be checked against requirements stated in natural language (e.g., policies, guidelines, quality criteria, or institutional expectations). Such requirements are easy to state but difficult to implement/audit.

The problem is acute in higher education. Course materials do more than describe a course; they make intended learning outcomes, teaching activities, assessment expectations, and students' responsibilities explicit. This view of course materials as explicit design artifacts aligns with constructive alignment, which treats outcomes, instruction, and assessment as one design problem [2], and with work on educative curriculum materials [11].

Many such requirements cannot be reduced to mechanical checks. A reviewer may need to judge whether assessment criteria are sufficiently explicit to guide students, whether feedback opportunities are meaningfully built into the course, or whether a policy provides students with a clear basis for action. These judgments must be grounded in the document's content. However, they also require interpreting how that evidence functions in context — a challenge that grows when requirements are course- or institution-specific or tied to a particular pedagogical approach.

Recent work has begun to use Large Language Models (LLMs) to support the evaluation and design of educational materials. Studies of document-level pedagogical evaluation target lesson plans or K-12 instructional materials, where models prove more reliable on explicit, textually grounded features than on high-inference pedagogical criteria [17,20]. In higher education, related work examines LLM support for generating and evaluating course materials, including syllabi, assessments, and slides [35]. However, in most model-assisted review settings, criterion interpretation, evidence selection, and final judgment remain coupled in a single model-mediated process [4]. This entanglement makes the review hard to audit: when a verdict is wrong, it is difficult to tell whether the failure stems from misunderstanding the requirement, selecting weak evidence, or applying an inconsistent decision rule.

To make such reviews auditable, we present AiDER[4], a Small Language Model (SLM) framework that separates interpretation from judgment. AiDER requires each requirement to be represented as a rule in a restricted domain-specific language (DSL) based on the Common Expression Language (CEL) [7]. Static analysis of the compiled condition yields an evidence scaffold that asks the model to extract only the evidence needed to verify the rule. The model, therefore, does not decide whether the requirement is satisfied; given the evidence, the binary verdict follows deterministically from executing the compiled condition. The resulting trace is inspectable by construction (each requirement to its rule, each rule to its evidence, and each verdict to that evidence), and fail-

[4] Source code and replication materials are available at `https://github.com/HISTLab-hevs/AiDER`.

ures are easier to diagnose, since errors in compiling rules, extracting evidence, and reaching a verdict can be analyzed separately.

We target 2B–4B class models that can be deployed on-premises. Educational materials are valuable intellectual property, and institutional policies may forbid sharing internal documents with third-party cloud LLMs. Although we evaluate AiDER in education, the pipeline is designed for auditable, requirement-based document review more broadly; however, its transferability to other domains is not evaluated in this study. Since these models may generate non-executable CEL rules, AiDER uses compilation errors as feedback in a repair loop to iteratively correct invalid rules. In turn, our work characterizes the reliability of rule compilation through three aspects: (a) the SLMs' ability to compile natural-language requirements into executable and sound CEL rules; (b) the effect of validation-driven repair on rule validity and fidelity; and (c) the impact of the resulting rules on document-level verdicts, including the contribution of model-generated extraction guidance.

To study these aspects, we instantiated AiDER on 200 applicable requirement-document pairs from instructional PDFs, comparing human reference rules, direct LLM-compiled rules, and validation-loop repair to trace how executable formalizations affect document-level verdicts. We also conducted a scaffold-only ablation to isolate the contribution of model-generated extraction guidance beyond the statically derived evidence scaffold.

The remainder of the paper is organized as follows: Section 2 reviews related work; Section 3 presents the methodology; Section 4 reports the results; and Sections 5 and 6 discuss limitations and conclude the paper.

2 Related Work

Recent works explore whether LLMs can evaluate educational artifacts against pedagogical criteria. Hauk and Soujon [17] analyze written lesson plans and show the difficulty of obtaining reliable judgments on criteria that require pedagogical interpretation beyond explicit document features, while Li et al. [20] introduce SciEval, where models evaluate K-12 science instructional materials by producing rubric-aligned scores and supporting evidence. In higher education, Yao et al. [35] examine LLM-generated course materials, including syllabi, assessments, and slides, showing that course-material quality can itself become an object of model-supported evaluation. Broader work on LLM-assisted educational assessment also emphasizes the importance of explicit rubrics, operational criteria, and human refinement in model-supported evaluation workflows [12,6]. Across these settings, a recurring limitation is that model-based evaluation often couples evidence identification, criterion interpretation, and verdict assignment in a single model-mediated step.

The same concern appears in legal and compliance-oriented document review, where recent systems move beyond direct document-level prompting by introducing structured intermediate steps. Narendra et al. [26] propose a template-

to-contract comparison pipeline that extracts key concepts from legal templates and checks their entailment in contracts through NLI/LLM-based reasoning. Sun et al. [31] propose a RAG-based compliance-checking framework that combines layered reasoning with eventic graphs for regulatory information. These approaches show that structuring review before assigning a decision has benefits, but their decisions still rely on entailment or retrieval-augmented reasoning over intermediate representations. AIDER takes a different approach: whether the requirement is given directly as a CEL rule or compiled from natural language, the rule defines the evidence scaffold and the decision logic. The model extracts structured, traceable evidence, but the final verdict is produced by deterministic rule evaluation.

Our approach also relates to semantic parsing and neuro-symbolic AI. Semantic parsing frames the problem as mapping natural language into executable logical forms, making interpretation checkable through formal representations [21]. Recent LLM-based systems extend this idea by translating natural-language problems into symbolic programs while relying on deterministic execution for inference. For example, Logic-LM translates reasoning problems into symbolic formulations and uses a solver to perform the final inference [29], while Crocetti and Dragoni [9] translate natural-language problems into executable Prolog [18] code and refine invalid programs through execution feedback. Similarly, Mensfelt et al. [25] introduce PrologMCP, which exposes Prolog through the Model Context Protocol (MCP)[5] as a reusable, stateful tool interface for LLM agents, supporting solver-backed inference through a translate-run-inspect-repair loop. More broadly, survey work on hybrid AI motivates combining data-driven and rule-based components to support reliability and transparency, while noting that evaluation and explanation practices remain fragmented [28]. AIDER integrates these principles into an auditable pipeline for requirement-based document review under a small-model constraint.

3 Methodology

AIDER replaces the single opaque verdict with a chain of inspectable steps. The remainder of this section first formalizes the review task, then describes the pipeline and compilation methods, and finally instantiates the setup in the higher-education case study.

3.1 Task Formalization

Let $\mathcal{R} = \{r_i\}_{i=1}^{m}$ be a set of pedagogical requirements, $\mathcal{D} = \{d_j\}_{j=1}^{n}$ a corpus of instructional documents, and $\mathcal{T} = \{t_h\}_{h=1}^{q}$ the set of document types, such as slides, assessments, syllabi, and rubrics. The functions θ and γ associate each requirement r_i with its single applicable type $\theta(r_i) \in \mathcal{T}$, and each document d_j with its non-empty set of document types $\gamma(d_j) \subseteq \mathcal{T}$, respectively. Each document is represented as an ordered page sequence (Definition 1).

[5] `https://modelcontextprotocol.io/docs/getting-started/intro`

$$d_j = (p_{j1}, p_{j2}, \ldots, p_{j\ell_j}) \tag{1}$$

The review task is restricted to type-compatible requirement-document pairs (Definition 2).

$$\mathcal{A} = \{(r_i, d_j) \in \mathcal{R} \times \mathcal{D} \mid \theta(r_i) \in \gamma(d_j)\} \tag{2}$$

For each $(r_i, d_j) \in \mathcal{A}$, the goal is to assign a binary verdict $y_{ij} \in \{0, 1\}$, where $y_{ij} = 1$ indicates that d_j satisfies r_i, and $y_{ij} = 0$ indicates that it does not. Pairs outside $\mathcal{A}$ are not evaluated.

3.2 Pipeline Architecture

AiDER operates in five phases illustrated in Figure 1. Phases PH1, PH3, and PH4 involve LLM inference, while PH2 and PH5 are deterministic. In PH1, the model compiles each natural-language requirement r_i into an executable CEL rule, combining all sub-requirements into a single compound condition using the evidence functions in Table 2. In PH2, the compiled rule is statically analyzed to recover all evidence item kinds and field paths without any LLM calls, producing an *evidence scaffold* that bounds what the extractor may emit. In PH3, the model reads the scaffold and produces per-kind and per-field *extraction instructions* that are constrained to the scaffold's kinds and fields. Together, the scaffold and these instructions form the *extraction specification*. In PH4, each document d_j is processed in page batches, and the model returns structured *evidence items* carrying a kind, a verbatim evidence quote, and a field map. In PH5, the aggregated evidence items are evaluated deterministically against the compiled CEL rule, yielding the binary verdict $y_{ij} \in \{0, 1\}$ without an LLM call.

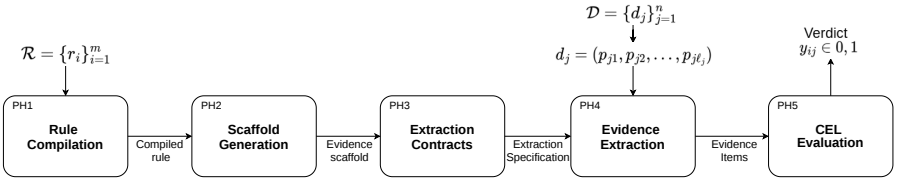

Fig. 1. Overview of the AiDER pipeline.

3.3 Rule Compilation Methods

Figure 2 illustrates the three methods for obtaining CEL rules. Let ρ_i denote a compiled rule for requirement r_i, and let $\text{valid}(\rho_i) \in \{0, 1\}$ denote the deterministic validation procedure that checks whether the generated rule is syntactically well-formed and compatible with the restricted rule interface.

Under *Reference rules*, the human-written rule $\rho_i^\star$ associated with each requirement r_i is loaded directly and used as the reference formalization against which generated rules are assessed.

Under *Direct compilation*, the model translates each natural-language requirement into a candidate rule in a single attempt (Equation 3):

$$\tilde{\rho}_i = \mathrm{LLM}_{T_0}(r_i) \tag{3}$$

Equation 4 represents the resulting compiled rule:

$$\hat{\rho}_i^{\mathrm{direct}} = \begin{cases} \tilde{\rho}_i, & \text{if valid}(\tilde{\rho}_i) = 1 \\ \bot, & \text{otherwise} \end{cases} \tag{4}$$

where $\bot$ denotes a compilation failure.

Under *Validation-loop compilation*, invalid rules enter a bounded repair loop initialized with $\rho_i^{(0)} = \tilde{\rho}_i$. At iteration k, the model is given the previous invalid rule and the corresponding validation error, and produces a repaired candidate as illustrated in Equation 5:

$$\rho_i^{(k)} = \mathrm{Repair}_{T_k}\big(r_i, \rho_i^{(k-1)}, \epsilon_i^{(k-1)}\big) \tag{5}$$

where $\epsilon_i^{(k-1)}$ is the validation error from the previous attempt. The model temperature is raised across repair attempts as $T_k = T_0 + k \cdot \Delta T$, with $k = 1, \ldots, 5$, $\Delta T = 0.1$, resulting in repair temperatures from 0.1 to 0.5 [9]. The loop returns the first valid repaired rule (if one exists) or $\bot$ otherwise.

3.4 Dataset and Rule Corpus

We evaluated AIDER on open-access *Business Communications 2* course pack [24], a first-year undergraduate course at Coast Mountain College. The course spans 13 weeks and covers professional communication strategies, including bad-news delivery, informational reports, persuasive messages, and presentation design. The corpus contains 15 unique PDFs. Because some PDFs belong to multiple document types, the applicable-type counts are 8 slide decks, 6 assessment documents, 5 rubrics, and 1 syllabus.

We defined 40 human-written rules, distributed evenly across the four document types, to serve as a reference formalization. The rules' design was inspired by literature on AI-aware assessment design [8,22], peer instruction and active learning [10,13], syllabus design and tone [16,30,5], constructive alignment [2], formative assessment and feedback [27,14], cognitive-load reduction in multimedia learning [23], and rubric design [3]. The resulting benchmark set is designed to cover common pedagogical concerns while exercising a diverse range of condition structures.

For each reference rule, we also defined a paired natural-language requirement request, written in plain English and without formal notation. Table 1

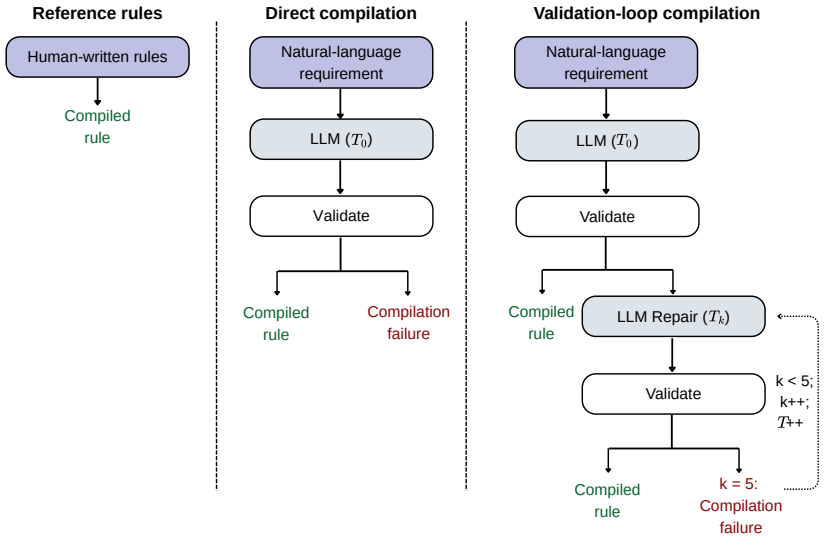

Fig. 2. Overview of the three compilation methods tested.

shows one reference rule and its paired natural-language request. Applicability is determined by matching each requirement to the corresponding document type. This type-matching procedure yielded $|\mathcal{A}| = 200$ applicable requirement-document pairs in the evaluated corpus.

Table 1. Example of a reference rule and its paired natural-language requirement.

Document type	Natural-language requirement	Reference rule
Assessment	At least one grading criterion should be stated explicitly enough that students understand how their work will be evaluated.	`item_count_where("grading_criterion", "normalized.is_explicitly_stated", "==", "true") >= 1`

AiDER's rule language is deliberately restricted to a fixed CEL interface. The interface exposes seven custom functions over extracted evidence items, covering item counts, existence checks, distinct-value counts, filtered counts and ratios, monotonic ordering checks, and group-level coverage checks. Human-written reference rules and model-compiled rules are therefore expressed through the same executable vocabulary. Table 2 lists the supported signatures.

This fixed set of functions makes each rule statically inspectable. Since evidence item kinds and field paths appear explicitly as function arguments, they can be recovered deterministically from the CEL condition. These elements de-

Table 2. The seven custom functions in AIDER. `kind` selects the evidence kind; `path` (and `o_path`/`v_path` the ordering/value keys, `grp`/`fp` the grouping/filter keys) is a field path that must begin with `normalized.`; `op`/`val` form a filter predicate `field op val`; `dir` ∈ {`non_decreasing`, `strictly_increasing`}; supported operators: `==`, `!=`, `>`, `>=`, `<`, `<=`, `in`, `not_in`.

#	Signature	Returns
1	`item_count(kind)`	`int`
2	`item_exists(kind)`	`bool`
3	`item_distinct_count(kind, path)`	`int`
4	`item_count_where(kind, path, op, val)`	`int`
5	`item_ratio_where(kind, path, op, val)`	`float`
6	`item_ordered(kind, o_path, v_path, dir)`	`bool`
7	`item_all_groups_have(kind, grp, fp, op, val)`	`bool`

fine an evidence scaffold that constrains what the model must retrieve before any extraction instructions are generated.

4 Results

We evaluated four models under the intentional 2B–4B parameter constraint: GEMMA-4-E4B-IT, GEMMA-4-E2B-IT [15], QWEN3.5-2B, and QWEN3.5-4B [34]. Each model was assessed under the three configurations reported in Figure 2. Each model was run three times; all tables report mean ± sample standard deviation over these runs. All inference calls in the AIDER pipeline were served locally with vLLM [19], on a single NVIDIA DGX Spark workstation[6].

All models were run with sampling temperature $T_0 = 0$ and *constrained decoding* enabled. Constrained decoding restricts token sampling at each step to tokens permitted by the target JSON schema, so the model can emit only output that conforms to the expected structure [32]. The validation error returned to the repair loop (Section 3.3) is therefore always a CEL compilation error.

We turned off thinking mode across all runs. In preliminary experiments, the smaller Qwen models, in particular, tended to enter reasoning loops when thinking was active, exhausting the token budget within the `<think>` block and returning empty content.

For document-level evaluation, we employed Claude Opus 4.8 [1] as a fixed external LLM judge, following the LLM-as-a-judge evaluation protocol [36]. The judge is not part of the AIDER pipeline; it was used only to obtain a stable reference signal for evaluation. For each applicable pair, the judge received the document, the human reference rule description, and the condition. The resulting reference verdict was therefore anchored to the reference formalization and held constant across all runs. Consequently, generated rules may be penalized when they encode a reasonable alternative interpretation that differs from the human-written rule. Since the latter is fixed, differences across models in the reference-rule condition reflect extraction quality rather than variation in rule content.

[6] `https://www.nvidia.com/en-us/products/workstations/dgx-spark/`

Evaluation metrics. From the rule perspective, we report the validation pass rate and three fidelity metrics relative to the human-reference rule. Evidence-kind Jaccard measures overlap in the evidence item kinds used by the two rules. Clause F1 compares executable structure by decomposing CEL conditions into clauses and scoring each clause by its best-matching counterpart in the other condition, using the mean Jaccard over rule function, evidence kind, and field paths. Cosine similarity, computed with BGE-small-v1.5 [33], captures semantic proximity but not executable equivalence. Failed compilations received a score of 0 on all rule-fidelity metrics. From the document perspective, we report Cohen's κ and Matthews Correlation Coefficient (MCC) against the fixed judge reference over all applicable requirement-document pairs. When compilation failed, and no verdict could be produced, the missing verdict was treated as maximal disagreement with the judge by assigning the opposite label. Negative agreement values therefore reflect both wrong verdicts and unrecovered compilation failures under this penalty. In the extraction-guidance Ablation, $\Delta\kappa$ and ΔMCC denote the gain of the complete extraction specification over the scaffold-only setting. Together, these metrics allowed us to analyze the three aspects outlined in Section 1.

Compilation quality

Table 3 reports compilation along two axes: whether a rule executes (Valid%) and how closely it matches the reference (Kind J., Cl. F1, Cosine). Under direct compilation, the 4B models compile most rules correctly, but validity and fidelity differ by model. Gemma-4-E4B-it leads on both, whereas Qwen3.5-4B approaches its validity yet trails sharply on evidence-kind Jaccard: its rules execute but rest on different evidence. The ordering Cosine > Kind J. $\geq$ Cl. F1 holds for every model, since cosine rewards rough semantic agreement while Kind J. demands an exact match of evidence kinds. A valid rule that reads closely to the reference can still select the wrong evidence, which is the component that determines what a rule actually checks and whose effect surfaces in the document-level results.

Table 3. Rule-compilation quality under direct and validation-loop compilation.

	Direct compilation				*Validation-loop compilation*			
Model	Valid%	Kind J.	Cl. F1	Cosine	Valid%	Kind J.	Cl. F1	Cosine
Gemma-4-E4B-it	**86.7**$\pm$3.8	**.611**$\pm$.05	**.408**$\pm$.02	**.761**$\pm$.04	**95.8**$\pm$1.4	**.702**$\pm$.02	**.462**$\pm$.01	**.841**$\pm$.01
Qwen3.5-4B	79.2$\pm$1.4	.412$\pm$.00	.326$\pm$.00	.701$\pm$.01	95.0$\pm$0.0	.438$\pm$.00	.365$\pm$.00	.840$\pm$.00
Gemma-4-E2B-it	36.7$\pm$1.4	.108$\pm$.01	.104$\pm$.01	.315$\pm$.01	56.7$\pm$2.9	.250$\pm$.00	.188$\pm$.01	.490$\pm$.03
Qwen3.5-2B	67.5$\pm$2.5	.216$\pm$.06	.212$\pm$.02	.582$\pm$.02	89.2$\pm$2.9	.320$\pm$.02	.294$\pm$.01	.772$\pm$.03

Table 4. Decomposition of compilation validity into first-attempt and repair-recovered rules, out of 40. Final valid equals initial valid plus recovered rules.

Model	Recovered	Unrecovered	F. Valid
Gemma-4-E4B-it	3.7±1.2	1.7±0.6	**38.3**±0.6
Qwen3.5-4B	6.3±0.6	2.0±0.0	**38.0**±0.0
Gemma-4-E2B-it	8.0±1.0	17.3±1.2	22.7±1.2
Qwen3.5-2B	8.7±0.6	4.3±1.2	35.7±1.2

Repair effects

The validation loop lifts both axes (Table 3, right): three of the four models exceed 89% validity, and fidelity rises across the board. The two 4B models, however, expose a limit of repair. Their cosine becomes nearly identical, while Qwen3.5-4B's evidence-kind Jaccard stays far below Gemma-4-E4B-it's (.438 vs .702): repair closes the semantic gap between them but not the gap in the evidence each rule requests, so Qwen's reliance on different evidence kinds is a stable trait rather than a repairable error. Table 4 locates the remaining failures: Gemma-4-E2B-it still leaves almost half of its initially invalid rules invalid after repair, whereas Qwen3.5-2B leaves only a small fraction. Repair, therefore, makes natural-language rules far more usable by recovering validity. Still, the evidence a rule targets is set at the first compilation and changes little afterward, which bounds how much repair can help downstream.

Document-level behavior

Table 5 reports agreement against the judge. The *Reference rules* column isolates extraction. It is the pipeline's strongest result: with a well-specified rule, three of the four models reach moderate to substantial agreement (up to $\kappa = .686$), and Gemma-4-E2B-it's score nearly matches Qwen3.5-4B's, so even a small model extracts evidence well once the rule is sound. Agreement drops for rules generated from natural language, and the negative values for the 2B models reflect the design penalty, since every failed compilation is scored as maximal disagreement. The two 2B models swap ranks between the settings: under reference rules Gemma-4-E2B-it leads (.534 vs .323) as the stronger extractor, but once the rule has to be created, the order reverses (.066 vs −.328), because Gemma-4-E2B-it's frequent compilation failures are penalized while Qwen3.5-2B compiles reliably enough to stay positive despite weaker extraction.

Ablation: scaffold-only versus guided extraction

The ablation evaluates whether LLM-generated extraction instructions provide useful guidance once the scaffold has already defined the admissible evidence types and fields. We compared the complete extraction specification with a scaffold-only variant, restricting the comparison to pairs associated with valid compiled rules under both extraction settings. Compilation failures were therefore excluded, so the values are not directly comparable to Table 5.

Table 5. End-to-end verdict agreement against the judge reference, pooled over 200 pairs.

	Reference rules		*Direct*		*Validation-loop*	
Model	κ	MCC	κ	MCC	κ	MCC
Gemma-4-E4B-it	**.686**$\pm$.04	**.697**$\pm$.04	**.246**$\pm$.04	**.246**$\pm$.04	**.336**$\pm$.03	**.344**$\pm$.03
Qwen3.5-4B	.539$\pm$.03	.543$\pm$.03	.028$\pm$.05	.028$\pm$.05	.276$\pm$.03	.282$\pm$.03
Gemma-4-E2B-it	.534$\pm$.01	.559$\pm$.01	$-$.504$\pm$.05	$-$.506$\pm$.05	$-$.328$\pm$.12	$-$.336$\pm$.12
Qwen3.5-2B	.323$\pm$.03	.323$\pm$.03	$-$.228$\pm$.08	$-$.228$\pm$.08	.066$\pm$.04	.073$\pm$.05

Table 6. Extraction guidance ablation: per-method gain from LLM-generated extraction contracts vs. the judge reference.

	Ref. rules		*Direct*		*Repair*	
Model	$\Delta\kappa$	ΔMCC	$\Delta\kappa$	ΔMCC	$\Delta\kappa$	ΔMCC
Gemma-4-E4B-it	**.045**$\pm$.04	**.040**$\pm$.03	**.085**$\pm$.05	.058$\pm$.06	.038$\pm$.02	.013$\pm$.02
Qwen3.5-4B	.001$\pm$.02	$-$.002$\pm$.02	.035$\pm$.05	.035$\pm$.04	.001$\pm$.01	.000$\pm$.01
Gemma-4-E2B-it	$-$.039$\pm$.01	$-$.040$\pm$.01	.052$\pm$.14	**.089**$\pm$.19	.058$\pm$.01	**.122**$\pm$.05
Qwen3.5-2B	$-$.181$\pm$.03	$-$.194$\pm$.03	.080$\pm$.13	.086$\pm$.14	**.078**$\pm$.03	.085$\pm$.02

The scaffold already captures much of the extraction intent. Additional instructions add little to human-written reference rules, where the scaffold is well-specified, and can even hurt weaker models such as Qwen3.5-2B. Their benefit is clearer with natural-language-compiled rules, where the scaffold is derived from imperfect model-generated conditions, leaving more room for disambiguation. The pattern is therefore consistent across methods: the statically derived scaffold carries most of the extraction intent on its own, and model-generated extraction contracts pay off mainly when the underlying rule is weak or underspecified. Once the scaffold is already well-formed, they become redundant or even counterproductive.

5 Limitations

The results of this study should be read in light of the evaluation's scope and design choices. The evaluation was designed as a focused case study in one higher-education setting. It used one course pack, with 15 PDFs, 40 requirements, and 200 applicable requirement-document pairs. This setting was useful for studying AiDER's behavior in detail, but it does not demonstrate how the framework would perform across many courses, institutions, or document types. Future work should therefore test the approach on larger and more diverse collections.

The human-written CEL rules served as reference rules, but they are not the only way to express each requirement. Indeed, some pedagogical requirements can be interpreted in more than one reasonable way. For this reason, the rule-level metrics indicate how close the generated rules are to our reference rules, rather than proving that a single rule is the only correct interpretation. Similarly, the document-level judge provides a consistent reference signal, but its proprietary

nature limits strict reproducibility; future work should therefore include expert human review.

Overall, AIDER reduces the role of model-based judgment, but it still depends on LLM outputs in some parts of the pipeline. In particular, the model is used for rule compilation, extraction guidance, and evidence extraction. Errors in these steps can still affect the final verdict, even though the verdict itself is computed deterministically. The rule interface is also intentionally limited to keep the system inspectable and easy to validate. Extending the interface while preserving auditability and benchmarking runtime remain future work.

6 Conclusion

We introduced AIDER, a framework for auditable, requirement-based document review, which we evaluated here on teaching materials under a small-model constraint. Its central design choice is to separate interpretation from judgment: requirements are compiled into executable CEL rules, rules induce bounded evidence scaffolds, and verdicts are obtained through deterministic rule execution over extracted evidence.

The results show that syntactic validity, semantic fidelity, and document-level agreement are distinct dimensions of the same auditability problem. The 4B models can already produce valid rules, but these may still define a different evidence scaffold or decision boundary from the intended reference formalization. Given a well-specified rule, even small models can extract sufficient evidence to produce reliable deterministic verdicts, which is the clearest sign that the design works. Validation-driven repair improves validity and fidelity jointly, yet its end-to-end benefit is scale-dependent: the 4B models recovered measurable document-level agreement, whereas the 2B models remained limited even after repair. At the document level, the binding constraint for generated rules is therefore rule semantics rather than executability. Finally, the extraction-guidance ablation shows that the statically derived scaffold already encodes most of the extraction intent, with model-generated guidance adding value primarily when the compiled rule is imperfect.

Overall, the findings support treating pedagogical review as a formalization and evidence problem rather than as a direct model-judgment task. AIDER does not remove the need for expert rule design or human oversight. Still, by making rules and evidence inspectable, it traces each failure to a specific stage rather than to a single opaque verdict. Future work should scale the evaluation to more courses and institutions, complement LLM-based judgment with expert human review, explore heterogeneous setups that pair a stronger model for compilation with a small model for extraction, and extend the DSL without compromising static analysis.

Acknowledgments. This work was supported by the HES-SO R&I Mind the Gap: Adapting Higher Education to the Age of AI (MTG, 140012/RI-PSTRAT25-06), HES-SO RCSO ISNet Hybrid Ai foR RelIable perSonalized cOachiNg (HARRISON) grant, Swiss AI Initiative (a0138).

References

1. Anthropic: Claude opus 4.8 system card (May 2026), https://www.anthropic.com/claude-opus-4-8-system-card, model system card; accessed 2026-06-02
2. Biggs, J.: Enhancing teaching through constructive alignment. Higher Education **32**(3), 347–364 (Oct 1996). https://doi.org/10.1007/BF00138871
3. Brookhart, S.M.: How to create and use rubrics for formative assessment and grading. ASCD, Alexandria, VA (2013)
4. Chiang, C.H., Chen, W.C., Kuan, C.Y., Yang, C., Lee, H.y.: 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. pp. 2489–2513. Association for Computational Linguistics, Miami, Florida, USA (Nov 2024). https://doi.org/10.18653/v1/2024.emnlp-main.146, https://aclanthology.org/2024.emnlp-main.146/
5. Chickering, A.W., Gamson, Z.F.: Seven principles for good practice in undergraduate education. AAHE Bulletin **39**(7), 3–7 (Mar 1987)
6. Chu, Y., Li, H., Yang, K., Copur-Gencturk, Y., Tang, J.: LLM-based automated grading with human-in-the-loop. In: 2025 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE). pp. 1–8. IEEE, Macao (Dec 2025). https://doi.org/10.1109/TALE66047.2025.11346771
7. Common Expression Language: Common expression language. https://cel.dev/ (2026), accessed 2026-05-25
8. Cotton, D.R.E., Cotton, P.A., Shipway, J.R.: Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. Innovations in Education and Teaching International **61**(2), 228–239 (2024). https://doi.org/10.1080/14703297.2023.2190148
9. Crocetti, V., Dragoni, A.F.: Bridging large language models and logic programming. In: 2025 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE). pp. 1066–1071. IEEE (2025). https://doi.org/10.1109/MetroXRAINE66377.2025.11340308
10. Crouch, C.H., Mazur, E.: Peer instruction: Ten years of experience and results. American Journal of Physics **69**(9), 970–977 (Sep 2001). https://doi.org/10.1119/1.1374249
11. Davis, E.A., Krajcik, J.S.: Designing educative curriculum materials to promote teacher learning. Educational Researcher **34**(3), 3–14 (Apr 2005). https://doi.org/10.3102/0013189X034003003
12. Emirtekin, E.: Large language model-powered automated assessment: A systematic review. Applied Sciences **15**(10), 5683 (2025). https://doi.org/10.3390/app15105683
13. Freeman, S., Eddy, S.L., McDonough, M., Smith, M.K., Okoroafor, N., Jordt, H., Wenderoth, M.P.: Active learning increases student performance in science, engineering, and mathematics. Proceedings of the National Academy of Sciences of the United States of America **111**(23), 8410–8415 (2014). https://doi.org/10.1073/pnas.1319030111
14. Gibbs, G., Simpson, C.: Conditions under which assessment supports students' learning. Learning and Teaching in Higher Education **1**, 3–31 (2005)
15. Google DeepMind: Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4 (2026), accessed 2026-05-28
16. Harnish, R.J., Bridges, K.R.: Effect of syllabus tone: Students' perceptions of instructor and course. Social Psychology of Education **14**(3), 319–330 (Sep 2011). https://doi.org/10.1007/s11218-011-9152-4

17. Hauk, D., Soujon, N.: 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**, 100538 (Jun 2026). https://doi.org/10.1016/j.caeai.2025.100538
18. Körner, P., Leuschel, M., Barbosa, J., Costa, V.S., Dahl, V., Hermenegildo, M.V., Morales, J.F., Wielemaker, J., Diaz, D., Abreu, S., Ciatto, G.: Fifty years of Prolog and beyond. Theory and Practice of Logic Programming **22**(6), 776–858 (Nov 2022). https://doi.org/10.1017/S1471068422000102
19. Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C.H., Gonzalez, J.E., Zhang, H., Stoica, I.: Efficient memory management for large language model serving with PagedAttention. In: Proceedings of the 29th Symposium on Operating Systems Principles. pp. 611–626. SOSP '23, Association for Computing Machinery, Koblenz, Germany (2023). https://doi.org/10.1145/3600006.3613165
20. Li, Z., He, P., Chen, Z., Liu, H., Wang, Z., Li, T., Xiong, J.: SciEval: A benchmark for automatic evaluation of K–12 science instructional materials (2026). https://doi.org/10.48550/arXiv.2604.25472, https://arxiv.org/abs/2604.25472, arXiv preprint arXiv:2604.25472
21. Liang, P.: Learning executable semantic parsers for natural language understanding. Communications of the ACM **59**(9), 68–76 (Aug 2016). https://doi.org/10.1145/2866568
22. Lodge, J.M., Howard, S., Bearman, M., Dawson, P., Agostinho, S., Buckingham Shum, S., Deneen, C., Ellis, C., Fawns, T., Gniel, H., Harper, R., Henderson, M., Liu, D., Markauskaite, L., McLean, J., Perrotta, C., Schuwirth, L., Slade, C.: Assessment reform for the age of artificial intelligence. Tech. rep., Tertiary Education Quality and Standards Agency (Nov 2023), https://www.teqsa.gov.au/guides-resources/resources/corporate-publications/assessment-reform-age-artificial-intelligence, published 23 November 2023
23. Mayer, R.E., Moreno, R.: Nine ways to reduce cognitive load in multimedia learning. Educational Psychologist **38**(1), 43–52 (2003). https://doi.org/10.1207/S15326985EP3801_6
24. McMurray, K.: Business communications 2. Course pack, BCcampus Open Collection (2020), 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
25. Mensfelt, A., Prabhakaran, A., Haret, A., Trencsenyi, V., Stathis, K.: PrologMCP: A standardized prolog tool interface for LLM agents (2026). https://doi.org/10.48550/arXiv.2606.14935, https://arxiv.org/abs/2606.14935, arXiv preprint arXiv:2606.14935
26. Narendra, S., Shetty, K., Ratnaparkhi, A.: Enhancing contract negotiations with LLM-based legal document comparison. In: Proceedings of the Natural Legal Language Processing Workshop 2024. pp. 143–153. Association for Computational Linguistics, Miami, FL, USA (Nov 2024). https://doi.org/10.18653/v1/2024.nllp-1.11, https://aclanthology.org/2024.nllp-1.11/
27. Nicol, D.J., Macfarlane-Dick, D.: Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. Studies in Higher Education **31**(2), 199–218 (2006). https://doi.org/10.1080/03075070600572090
28. Pacioni, E., Coman, A.C., Calvaresi, D., Manzo, G., Schumacher, M.I.: Agent-based hybrid AI models and technologies: A systematic literature review. IEEE Access **14**, 21148–21166 (2026). https://doi.org/10.1109/ACCESS.2026.3661027

29. Pan, L., Albalak, A., Wang, X., Wang, W.: Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In: Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 3806–3824. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.findings-emnlp.248, https://aclanthology.org/2023.findings-emnlp.248/
30. Parkes, J., Harris, M.B.: The purposes of a syllabus. College Teaching **50**(2), 55–61 (2002). https://doi.org/10.1080/87567550209595875
31. Sun, J., Luo, Z., Li, Y.: A compliance checking framework based on retrieval augmented generation. In: Proceedings of the 31st International Conference on Computational Linguistics. pp. 2603–2615. Association for Computational Linguistics, Abu Dhabi, UAE (Jan 2025), https://aclanthology.org/2025.coling-main.178/
32. Willard, B.T., Louf, R.: Efficient guided generation for large language models (2023). https://doi.org/10.48550/arXiv.2307.09702, https://arxiv.org/abs/2307.09702, arXiv preprint arXiv:2307.09702
33. Xiao, S., Liu, Z., Zhang, P., Muennighoff, N., Lian, D., Nie, J.Y.: C-Pack: Packed resources for general chinese embeddings. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 641–649. Association for Computing Machinery (2024). https://doi.org/10.1145/3626772.3657878
34. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., Qiu, Z.: Qwen3 technical report (2025). https://doi.org/10.48550/arXiv.2505.09388, https://arxiv.org/abs/2505.09388, arXiv preprint arXiv:2505.09388
35. Yao, H., Xu, W., Turnau, J., Kellam, N., Wei, H.: 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 (Volume 1: Long Papers). pp. 4087–4109. Association for Computational Linguistics, Rabat, Morocco (Mar 2026). https://doi.org/10.18653/v1/2026.eacl-long.191, https://aclanthology.org/2026.eacl-long.191/
36. Zheng, L., Chiang, W.L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E.P., Zhang, H., Gonzalez, J.E., Stoica, I.: Judging LLM-as-a-judge with MT-Bench and chatbot arena. In: Advances in Neural Information Processing Systems 36. vol. 36, pp. 46595–46623. Neural Information Processing Systems Foundation (2023). https://doi.org/10.52202/075280-2020, https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html