Before Detection: Stress-Testing Engineering Controls Assessments with Large Language Model Red Teams

Dr. Alyson Grace Eggleston, The Pennsylvania State University

Alyson Eggleston is an Associate Professor in the Penn State Hershey College of Medicine and Director of Evaluation for the Penn State Clinical and Translational Science Institute. Her research and teaching background focuses on program assessment, STEM technical communication, industry-informed curricula, and educational outcomes veteran and active duty students.

Dr. Robert J. Rabb P.E., The Pennsylvania State University

Robert Rabb is the associate dean for education in the College of Engineering at Penn State. He previously served as a professor and the Mechanical Engineering Department Chair at The Citadel. He previously taught mechanical engineering at the United States Military Academy at West Point. He received his B.S. in Mechanical Engineering from the United Military Academy and his M.S. and PhD in Mechanical Engineering from the University of Texas at Austin. His research and teaching interests are in mechatronics, regenerative power, and multidisciplinary engineering.

Before Detection: Stress-Testing Engineering Controls Assessments with Large Language Model Red Teams

As generative AI tools become integrated into learning environments, educators must understand how different large language models (LLMs) interpret and solve assessment items intended to measure human reasoning. This study mimics potential student-user conditions by treating models as red team participants, allowing educators to probe the relative utility of AI systems in correctly identifying problem structure, articulating reasoning steps, and producing valid solutions at each level of Bloom's taxonomy. Comparative results reveal variation in problem-solving logic and in susceptibility to overproduced logic and solutions.

This study evaluates the performance of multiple LLMs, including Copilot, ChatGPT, and locally run Ollama instances, when tasked with answering Engineering Controls exam questions designed across Bloom's Taxonomy levels. A total of 21 exam items (short-answer and design-based) were drawn from Engineering Controls textbooks, test banks, and other sources. Each item was classified by independent reviewers according to Bloom's levels (remember, understand, apply, analyze, evaluate, create). Three LLM configurations were compared: (1) Copilot GPT-4 (cloud-hosted, proprietary weights); (2) ChatGPT-5.2 container; (3) Ollama Llama-3 70B (locally hosted, open-weights model). Models received no feedback between runs to avoid iterative tuning. Each model's response to every test item was scored using a rubric framework: Accuracy, Reasoning Transparency, and Bloom Alignment. The authors expect that higher order cognitive tasks will produce lower rated responses for AI reasoning and transparency.

This study offers practical value for educators by (1) identifying likely points of misinterpretation in exam questions, (2) revealing common solution patterns that emerge in AI-assisted responses, and (3) demonstrating how locally hosted, in-house LLMs can be used to audit assessment items and evaluate their robustness.

Keywords: Large Language Models (LLMs); Red-Team Evaluation; Bloom's Taxonomy; Assessment Design; Engineering Controls.

1. Introduction

Large language models (LLMs) offer a novel opportunity to evaluate educational assessments themselves by functioning as standardized, adversarial test-takers capable of systematically probing item structure, cognitive demand, and likely reasoning path. Early scholarship on generative machine learning in higher education has raised concerns that LLMs can generate fluent, plausible responses that obscure shallow reasoning and may challenge the utility of traditional assessment formats [1]. In parallel fields such as machine learning safety and software security, 'red-teaming' has emerged as an evaluative approach that treats intelligent systems as adversarial probes, deliberately stressing tasks to surface latent failures or weaknesses in content or logic, rather than relying only on prior performance metrics. Red Teaming is used in military planning and threat exercises for alternative analysis, simulation, and vulnerability probes [2]. Recent work demonstrates that models can be used instrumentally to expose weaknesses in other models, revealing sensitivities to problem structure, prompting, and underspecified constraints.

Drawing on this epistemology, we reframe LLMs as tools to mimic adversarial test-takers, allowing instructors to interrogate the resilience of engineering assessments under automation conditions. From this perspective, the vulnerability of interest is not model error per se, though its limits are explored, but instances in which assessment items permit superficially correct responses that bypass the intended cognitive process. By combining red-team evaluation with a Bloom's taxonomy–aligned analysis, this study positions assessment validity as a property that can be empirically stress-tested, offering a method for identifying which question types retain discriminative power for human reasoning in machine learning–mediated learning environments. Our work is guided by the following research questions, and explored in the context of a traditional engineering controls course and its major assessments.

Research Questions

(1) Which engineering controls assessment items are most solvable by LLMs?
(2) How does solvability vary by Bloom's taxonomy tier?
(3) How do the following LLM-derived solution features vary across models: answer length, correctness, conciseness

2. Background: Assessment Resilience

This approach draws conceptual grounding from red-teaming research in machine learning, which treats evaluation as a process of deliberately stressing the system to test the limits, rather than aggregate scoring or repeat performance measures. Perez [3] formalizes 'red teaming' language models using language models, demonstrating how adversarial probing can systematically elicit problematic behaviors and boundary conditions that remain invisible under conventional testing. In a related work, Perez and colleagues [4] show that models can generate evaluation items themselves with varying success to discover behavioral tendencies and failure modes in other models. Coder and colleagues use LLM performance to evaluate whether multiple choice questions are resistant to generative AI, finding that LLMs overselect for plausible, distractor answers, particularly in the context of higher order Bloom's tiers and drawing-based answers [5]. Our study extends this logic by treating LLM-based responses as diagnostic information for available exam item interpretations for exclusively open-ended questions, as well as a useful window into AI-assisted reasoning patterns to solution sets. Controlled, institutional environments are safer and secure intellectual property in a way that cloud-based third-party providers cannot guarantee. Leveraging Penn State's ROAR infrastructure, this workflow was operationalized with Ollama Llama-3 with a reach goal of creating an instrument that other educators can use to audit their examination items. Together, these works justify using LLMs not merely as objects of analysis, but as active instruments for generating evidence about task vulnerability, interpretive fragility, and sensitivity to prompt framing.

Work like this is made possible by industry documentation standards. Industry-facing documentation reinforces this epistemology by treating adversarial evaluation as a standard component of model assessment. OpenAI system cards (e.g., GPT-4) [6] describe internal and external red-team exercises that intentionally pressure-test models across risk areas, highlighting the value of structured adversarial protocols and systematic documentation of failure cases. While the safety goals differ from educational measurement, the methodological stance allows

for named stress conditions, structured elicitation, and documentation of failure modes, which are properties that align with the needs of assessment evaluation in machine learning-mediated learning environments.

Pursuing the Limits of Response Conditions
A parallel body of literature in software security positions LLMs as systems whose vulnerabilities can be investigated using systematic taxonomies of attacks, defenses, and evaluation methods. Surveys of LLMs in software security and vulnerability investigation [7] – [9] emphasize that performance alone is insufficient. Meaningful evaluation of test items requires identifying how specific inputs, task framings, and constraint structures elicit fragile responses, misinterpretations, or overconfident outputs. Translating this logic to education, the "vulnerability" of interest is an assessment design weakness: item properties that allow correct-looking responses without requiring the intended reasoning, or that collapse higher-order tasks into pattern-matching. These security-oriented taxonomies therefore provide a conceptual vocabulary for categorizing assessment types: ambiguity, missing constraints, cueing effects, and sensitivity to surface form.

Professional exam performance and the limits of correctness as evidence of reasoning
Work evaluating LLMs on professional credentialing assessments further motivates a construct-focused approach. Bommarito and colleagues [10], [11] show in a zero-shot evaluation, where no prior context or training is given, of CPA-related capabilities, that LLMs can achieve impressive outcomes on structured exam-like tasks while leaving open the question of what cognitive processes are being replicated or bypassed. For educational measurement, this distinction is critical: correctness may reflect retrieval, pattern completion, or heuristic matching rather than the cognitive depth an instructor intends to assess. As a result, evaluation frameworks must extend beyond accuracy to include features that approximate the actual process or reasoning path, ensuring that transparency and alignment between intended and demonstrated cognitive operations occurs under exam conditions. Notably, similar capabilities have been explored in the context of state bar exams for attorneys, with limited success especially for essay sections [12].

Bloom-aligned analysis as a pragmatic construct lens
Within engineering education, Bloom's taxonomy [13] remains an important lens for articulating intended cognitive demand and for distinguishing between recognition, procedural application, analytical decomposition, and design/evaluation judgment. In the present study, Bloom's levels are not treated as a claim about true cognition, but as a shared vocabulary for evaluating whether an assessment item's intended cognitive tier survives contact with LLM-mediated solution generation. By combining accuracy scoring with reasoning transparency and Bloom-alignment ratings, the study operationalizes a stress-test of assessment resilience**:** the degree to which different item types maintain differentiation power for human reasoning.

3. Methods

Study Design
We conducted a structured, prompt-standardized evaluation of Engineering Controls testing items using two Large Language Models (LLMs) as red team test-takers, ChatGPT-5.2 and CoPilot GPT-4. The unit of analysis was an exam item. For each item, a human-assigned

Bloom's Taxonomy cognitive tier served as the intended cognitive demand label. The LLM was then prompted under a fixed three-part protocol to (A) generate an exam-style solution, (B) self-classify the Bloom level evidenced by its own solution and justify that classification, and (C) report the token length of the Part A response. The study emphasized item vulnerability under automation pressure (e.g., items yielding correct-looking answers with limited reasoning transparency or collapsing higher-order prompts into lower-order solution behaviors).

Exam Sources

A set of 21 open-ended exam questions was drawn from a single Engineering Controls course assessment (mix of short-answer and design/problem-solving items). Items were compiled into an Excel-based schema to support standardized administration, metadata capture, and structured comparison across items. Each item was stored as a discrete record containing question text and item identifiers. All images used in assessment items were coindexed with item text. No item content was modified for the LLM other than insertion into the standardized prompt template.

Bloom Labeling

Prior to LLM testing, each exam item was rated for intended Bloom's Taxonomy tier [13] using the six-level cognitive process dimension: Remember, Understand, Apply, Analyze, Evaluate, Create by a Quality Matters-trained evaluator and a subject matter expert in engineering controls. Ratings were recorded in the Excel schema as the item's intended tier. This labeling was treated as a category identifier for an instructor's course design intent. It served as the reference label for comparing item intent to model-demonstrated behavior and self-classification.

LLM configuration and execution environment

The LLM was executed in a secure, temporary ChatGPT-5.2 container to minimize cross-run contamination and maintain consistent runtime conditions. No fine-tuning or training was performed for this study. The model received no feedback between items and no iterative prompt refinement occurred during the run. This approach supported a zero-feedback administration intended to approximate a fixed exam condition.

Prompting protocol (three-part structured prompt)

All items were administered using a fixed prompt with three sequential parts:

(1) **Part A. Solve the question.** The model was instructed to respond as a student taking an Engineering Controls exam, provide an exam-style solution, show reasoning and assumptions, and avoid external sources and narrative framing. The question text for each item was inserted into the template variable [Question Text]. The same template was used across all items to reduce prompt variance and isolate item-level differences.

(2) **Part B. Self-assessment of Bloom level.** Based only on its Part A response, the model selected a single Bloom tier (Remember/Understand/Apply/Analyze/Evaluate/Create) that it believed its solution demonstrated, and provided a brief justification.

(3) **Part C. Token length.** The model was instructed to report the token length of its Part A answer.

Data captured per item

For each item administration, the following outputs were captured and stored in the Excel:

a. LLM Part A response text. (solution output)

b. Time on task**.** (elapsed time from prompt submission to response completion)
c. LLM self-rated Bloom tier. (categorical selection)
d. Justification text for the self-rated Bloom tier
e. Token length of Part A (as reported by the model)
f. Human intended Bloom tier (pre-assigned)

Analysis Approach

Analyses were conducted at the item level. Primary comparisons included:

- Solvability by intended Bloom tier: distribution of accuracy and transparency scores across tiers.
- Bloom mismatch: cases where items labeled higher-order (Analyze/Evaluate/Create) yielded responses that appeared procedural or pattern-matched (Apply/Understand), including mismatches between intended tier and model self-rated tier.
- Efficiency: time on task and token length as indicators of response compression, verbosity, examined in relation to correctness and transparency.

Results were summarized using descriptive statistics and comparisons.

4. Results

Preliminary results suggest a straightforward divergence between intended Bloom categorization and the LLM-assigned Bloom classification. Across 21 assessment items, LLM responses tended to be categorized as *Analyze*, regardless of the intended Bloom tier. Items that were originally designed to assess lower order cognitive processes (*Remember, Understand, Apply*) were frequently identified as higher-order processes (*Analyze*). In parallel, items designed to elicit higher Bloom's tiers (*Evaluate, Create*) were consistently classified as intermediate-level Bloom's tiers, resulting in a tendency toward the middle of the scale. Taken together, these patterns suggest that LLM-generated answers may overengineer the solutions needed to satisfy an assessment item, while also compressing the features needed to produce a higher-order answer. Table 1 shows the distribution of assessment items, their Instructor-assigned Bloom's level, and the average token length of responses corresponding to the instructor-assigned items. Items with asterisks indicate assessment items for which we currently do not have estimates under test-taking conditions, but the authors look to include those in a revision to this work. Token, as captured in Tables 1-2, represents a basic unit of text or meaning processed by an LLM, and can represent a word, affix, punctuation, or equation variables, as defined by the individual model tokenizer [14].

Table 1: Distribution of controls assessment items among instructor-assigned Bloom's levels and average token length

Blooms Level **(Instructor Defined Level)**	**# Questions classified**	**Human - Average time to complete / answer question (seconds)**	**Token Length per Human-Assigned Tier**
Remember	2	75	151
Understand	1	*	29
Apply	5	293	242
Analyze	9	212	253
Evaluate	1	204	110
Create	3	*	480
Total	**21**		

Token counts reflect the length of a response output in model-referential units rather than typical units, (e.g., word count). The instructor had taught the course for 12 years, often each semester, and was very familiar with the material, hence, the quick response times per item. Notably, despite this apparent collapse toward the middle of Bloom's taxonomy, the LLM produced significantly larger-token counts in responding to *Create*-level items. Showing the collapsed distribution, Table 2 provides additional time-to-completion data per LLM-assigned Bloom level.

Table 2: Distribution of controls assessment items among LLM-assigned Bloom's levels, task time, and token length

Blooms Level **(LLM-Defined Level)**	**# Questions Classified**	**LLM - Average time to complete / answer (seconds)**	**Token Length – Average per Tier**
Remember	1	15.0	62
Understand	0	NA	220
Apply	7	38.9	NA
Analyze	13	67.3	316
Evaluate	0	NA	NA
Create	0	NA	NA
Total	**21**	AVERAGE 55.3	AVERAGE 272

To examine whether Bloom drift reflected differences in model effort or output token length, we analyzed correlations between intended Bloom level, observed Bloom classification, time on task, and response token length. Spearman's rank-order correlation was used because Bloom's taxonomy represents an ordinal scale and because response time and token distributions violated normality assumptions. Spearman correlation analyses showed that the intended Bloom level was significantly associated with response token count ($\rho = 0.46$, $p = 0.038$), and moderately but not significantly associated with LLM time-on-task. In contrast, the Bloom 'drift', or difference between the instructor-assigned Bloom level and the LLM-categorized level, was not significantly correlated with token count or response time.

These findings suggest that the relationship between instructor-intended assessment item complexity and LLM-generated answers is predictive of much longer answers, despite the LLM's apparent blindness to categorizing assessment items by their complexity.

Table 3: Spearman correlations between intended Bloom's level, LLM effort, and Bloom drift

Predictor	**Outcome**	**ρ (Spearman)**	***p***
Intended Bloom	Token Count	**0.46**	**0.038**
Intended Bloom	LLM Time	0.41	0.067
Bloom Drift	Token Count	−0.25	0.280
Bloom Drift	LLM Time	−0.16	0.486

Case Example - Exam 1 Problem 1
Two case examples are selected for review because they provoked extended processing times (90 and 48 seconds, respectively) and significantly longer answers (650 and 230 tokens, respectively).

The following example was selected to compare the instructor and LLM-derived solutions, and offer notes on disparities. For figure 1, students were asked to "Redraw the circuit in the Laplace Domain (5 points)." LLM responses included unnecessary information (e.g., impedance elaborations) and tended to overcomplicate otherwise straightforward solutions. While resistor and inductor representations and the time-to-Laplace transformation were handled correctly, the capacitor term was consistently misrepresented, with s incorrectly placed in the numerator (e.g., "S100" instead of 100/s). Additional issues included inappropriate simplifications (e.g., invalid "collapse" steps). Overall performance was partially correct, with estimated scores in the 70–80% range (≈4/5), reflecting strong procedural structure but key conceptual errors.

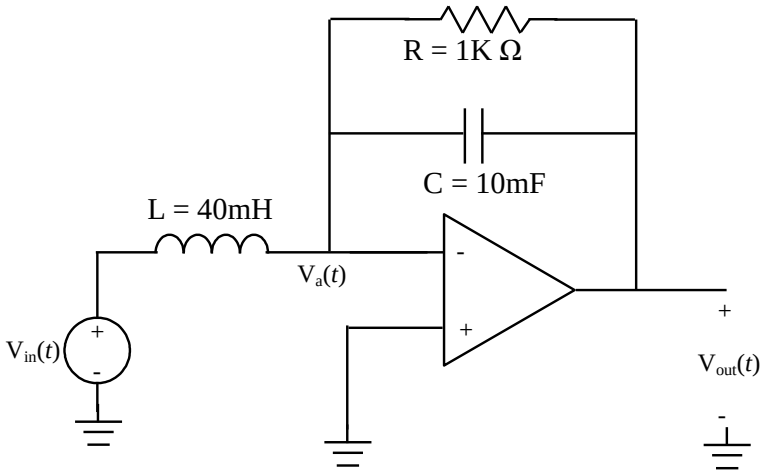


Figure 1: Laplace diagram

ChatGPT Solution

Element substitutions

- Inductor L = 40 mH → impedance
  ZL(s) = sL = s(40×10−3) = 0.04s Ω
- Capacitor C = 10 mF → impedance
  ZC(s) = 1/sC = 1/s(10×10−3) = 1/0.01s = 100/s Ω
- Resistor R = 1 kΩ → impedance
  ZR(s) = 1000 Ω
- Source vin(t) → Vin(s), output vout(t) → Vout(s)

Optional: combine the feedback impedance into one block

Since the resistor and capacitor are in parallel between Vout and Va,

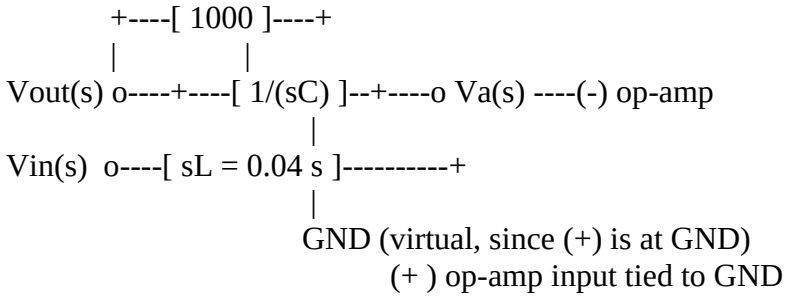
$$Z_f(s) = Z_R \parallel Z_C$$
$$= 1 / (1/R + 1/Z_C)$$
$$= 1 / (1/R + sC)$$
$$= R / (1+sRC)$$

With R = 1000 Ω, C = 0.01 F,

$$Z_f(s) = 1000 / (1+10s)\ \Omega$$

So, an even cleaner Laplace redraw is the same schematic, but replacing the parallel pair with a single impedance block Zf(s) = 1000 / (1+10s).

Zf(s) = R || 1/(sC)



Copilot Solution

Given circuit (assumptions):

- Ideal op-amp, G → ∞, input currents = 0.
- Initial conditions = 0.
- Laplace variable s.

(a) Circuit Redrawn in Laplace Domain

Replace each element by its Laplace impedance and signals by Laplace transforms:

- Input source:
  Vin(t) → Vin(s)
- Inductor L = 40 mH:
  ZL(s) = sL = 0.04s
- Resistor R = 1 kΩ:
  ZR = 1000
- Capacitor C = 10 mF = 0.01 F:
  ZC(s) = 1/sC = 1/0.01s = 100/s

Laplace-Domain Representation (textual)

- Vin(s) → series impedance sL → inverting node Va(s)
- Non-inverting input tied to ground:
  V+(s) = 0
- From output Vout(s) to inverting node Va(s):
- Resistor R = 1000
- Capacitor 1/(sC) = 100/s
- In parallel
- Ideal op-amp relation:
  V−(s) = V+(s) = 0 ⇒ Va(s) = 0

Compact Exam-Style Sketch (Laplace)

  - Vin(s) — sL —
- Va(s) = 0
- Feedback from Vout(s) to via
  R ‖ 1/sC
- Op-amp: ideal, referenced to ground.

For comparison purposes, here is the instructor-provided solution:

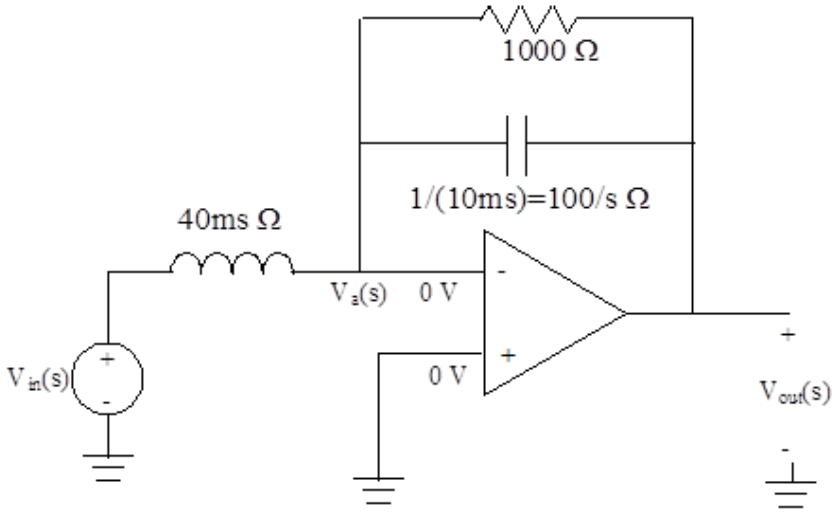


Figure 2: Redrawn instructor-provided solution for the Laplace domain

Case Example - Exam 1 Problem 5
In this problem, students were asked to “Find the transfer function for the system below (15 points).”

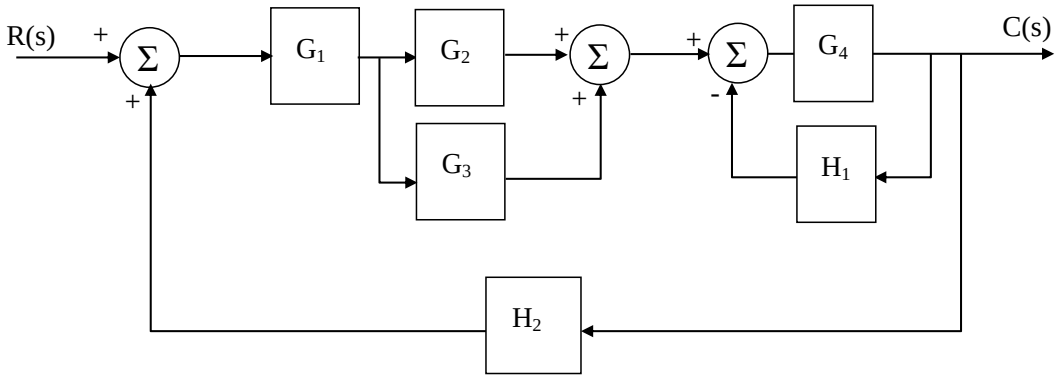


Figure 3: System diagram

On Exam 1 Problem 5, LLM performance was mixed: one model ChatGPT arrived at a correct intermediate form but introduced errors by over-extending the solution, while Copilot produced a mathematically correct final answer. Across both, responses were overcomplicated, with excessive and non-standard notation that made them difficult to interpret. Although the correct solution was sometimes embedded in the work, unnecessary transformations reduced. In practice, the instructor noted it would take less time to solve the problem manually than to decipher the LLM output, highlighting a key limitation in efficiency despite partial correctness.

ChatGPT Solution

1. Inner loop around G_4 with negative feedback H_1

$$C = G_4\,(e_2 - H_1 C)$$

$$C(1 + G_4 H_1) = G_4 e_2$$

$$C = \frac{G_4}{1 + G_4 H_1}\, e_2$$

2. Parallel G_2, G_3 fed by G_1

$$e_1 = R + H_2 C$$

$$x = G_1 e_1$$

$$e_2 = G_2 x + G_3 x = (G_2 + G_3) G_1 e_1$$

3. Substitute

$$C = \frac{G_4}{1 + G_4 H_1}(G_2 + G_3)G_1(R + H_2 C)$$

Let

$$K = \frac{G_4}{1 + G_4 H_1} G_1(G_2 + G_3)$$

Then:

$$C = KR + KH_2 C$$

$$C(1 - KH_2) = KR$$

$$\frac{C}{R} = \frac{K}{1 - KH_2}$$

Final expression

$$\frac{C(s)}{R(s)} = \frac{\dfrac{G_4 G_1(G_2 + G_3)}{1 + G_4 H_1}}{1 - \dfrac{G_4 G_1(G_2 + G_3)H_2}{1 + G_4 H_1}}$$

Equivalent simplified form:

$$\boxed{\frac{C(s)}{R(s)} = \frac{G_4 G_1(G_2 + G_3)}{(1 + G_4 H_1) - G_4 G_1(G_2 + G_3)H_2}}$$

Copilot Solution

Block Reduction

Forward path to first summing junction

Let the output of the first summing junction be:

$$E(s) = R(s) + H_2(s)C(s)$$

Through G_1:

$$X(s) = G_1(s)E(s)$$

Split:

- Upper path: $G_2(s)X(s)$
- Lower path: $G_3(s)X(s)$

Combine parallel paths

$$Y(s) = [G_2(s) + G_3(s)]\, G_1(s)\, [R(s) + H_2(s)C(s)]$$

Inner feedback loop with H_1

Second summing junction (negative feedback):

$$U(s) = Y(s) - H_1(s)C(s)$$

Through G_4:

$$C(s) = G_4(s)\,(Y(s) - H_1(s)C(s))$$

Solve inner loop:

Substitute $Y(s)$

$$C(s) = \frac{G_4}{1 + G_4H_1}(G_2 + G_3)G_1\left[R(s) + H_2C(s)\right]$$

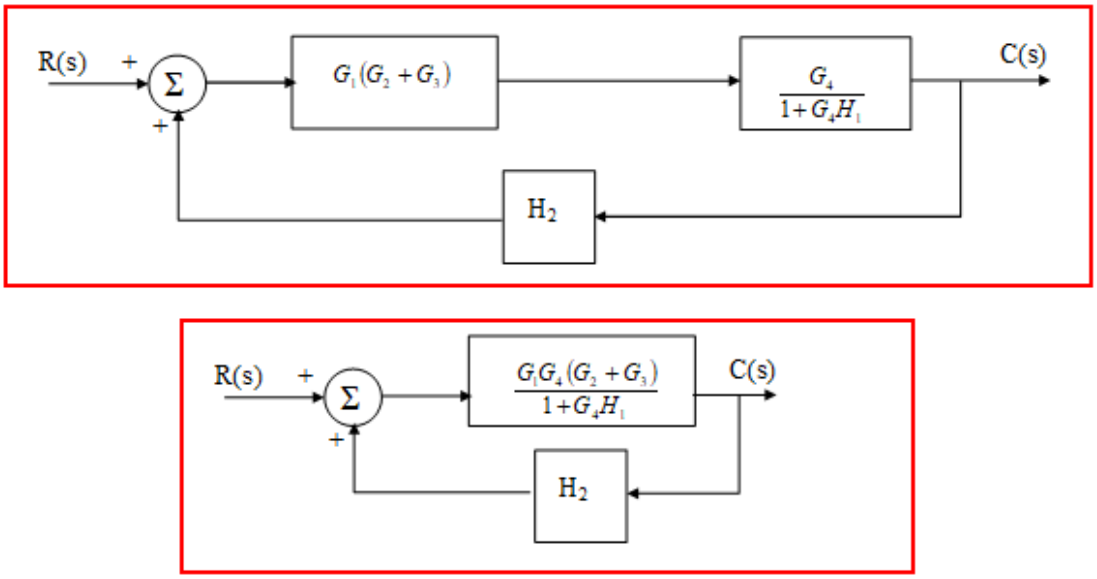
Solve for $\frac{C(s)}{R(s)}$

$$C(s)\left[1 - \frac{G_4(G_2 + G_3)G_1H_2}{1 + G_4H_1}\right] = \frac{G_4(G_2 + G_3)G_1}{1 + G_4H_1}R(s)$$

$$\boxed{\frac{C(s)}{R(s)} = \frac{G_1(G_2 + G_3)G_4}{(1 + G_4H_1) - G_1(G_2 + G_3)G_4H_2}}$$

Final transfer function

Figure 4 below represents the instructor solution using the textbook methodology and syntax.



$$\frac{C(s)}{R(s)} = \frac{\dfrac{G_1G_4(G_2+G_3)}{1+G_4H_1}}{1-H_2\dfrac{G_1G_4(G_2+G_3)}{1+G_4H_1}} = \frac{G_1G_4(G_2+G_3)}{1+G_4H_1-H_2G_1G_4(G_2+G_3)} = \frac{G_1G_4(G_2+G_3)}{1+G_4\left[H_1-H_2G_1(G_2+G_3)\right]}$$

Figure 4: Redrawn instructor-provided transfer function.

5. Discussion

These preliminary results suggest that without proper training and inputs including task-specific calibration, general purpose LLMs may be weak in instantiating model-categorical evaluative labels, even from a well-known construct like Bloom's taxonomy. Background knowledge of natural language processing and polysemy phenomena in human language would predict this result. Network-modeling studies of both human and LLM-language ascribing performance note that human semantic associations and their attested use differ from LLM semantic associations [15] – [17], which Bender and others have noted as a structural explanation for the apparent gap in LLM-produced responses that are interdisciplinary, lateral conceptualizations as compared to human researcher-derived responses [18]. This structural divergence from human language happens to also be a forensic 'water mark' of LLM-derived content.

We have already operationalized a locally hosted LLM environment using Ollama on Penn State's ROAR infrastructure, enabling secure, institutionally controlled evaluation workflows. While LLM-generated responses in this study are scored for correctness, reasoning transparency, and conciseness, these results also demonstrate a practical use case for education-focused LLM design. Specifically, task-specific calibration and structured prompting can be used to improve the consistency, interpretability, and utility of categorical classifications for assessment and measurement. In contrast to commercial models, which are optimized for general-purpose performance, locally deployed models can be tuned to align with domain-specific evaluation needs like Bloom's taxonomy or specific learning outcomes. Ongoing work extends this framework through the deployment of a Llama-3 model via Ollama to operationalize exam item auditing that other educators can also use for their own courses.

While the dataset currently includes limited exemplars for the higher order *Evaluate* and *Design* instructor-designating assessment items, current outputs indicate that general purpose LLMs can produce partially correct answers on these tiers, with co-occurring high token counts and relatively longer task times. Given the limited representation of these item types, subsequent analyses will shift focus from response characteristics to assessment item inputs, examining whether surface features, such as text-only, image-only, or combined text-and-image formats, are associated with systematic patterns in misalignment or problematic outputs. This secondary analysis aims to identify item-level affordances that interact with automated solution generation in ways that challenge construct instantiation.

This study is motivated not only by the development of assistive evaluation tools, but by emerging evidence that LLM responses surface underlying structural features of assessment items, including polysemy and uneven semantic density that can lead to patterned misinterpretation and overengineered responses. These findings underscore the need for institutionally controlled, domain-calibrated models rather than reliance on general-purpose commercial systems. We have begun operationalizing a locally hosted LLM environment to support this work, enabling task-specific prompting, constrained outputs, and reproducible evaluation workflows. Within this framework, LLMs can audit how assessment items are interpreted, classify cognitive demand (e.g., Bloom's levels), and map artifacts to program-level outcomes. In institutional contexts where intellectual property is prioritized and valued, a self-serve, locally deployed model offers instructors the ability to iteratively refine assessments,

identify potential ambiguities before deployment, and strengthen alignment between course-level instruments and program-level evaluation goals.

References

[1] J. Rudolph, S. Tan, and S. Tan, "ChatGPT: Bullshit spewer or the end of traditional assessments in higher education?," Journal of Applied Learning & Teaching, vol. 6, no. 1, pp. 342–363, 2023.

[2] M. Zenko, Red Team: How to Succeed by Thinking Like the Enemy. New York, NY: Basic Books, 2015.

[3] E. Perez et al., "Red teaming language models with language models," arXiv preprint arXiv:2202.03286, 2022.

[4] E. Perez et al., "Discovering language model behaviors with model-written evaluations," in Findings of the Association for Computational Linguistics: ACL 2023, pp. 13387–13434, 2023.

[5] J. B. Coder, J. G. Coder, and M. D. Maughmer, "Aerospace engineering education in the era of generative AI," in Proceedings of the 2025 ASEE Annual Conference & Exposition, Montreal, QC, Canada, Jun. 2025. doi: 10.18260/1-2—57581.

[6] OpenAI, "GPT-4 system card," 2024. [Online]. Available: https://cdn.openai.com/gpt-4o-system-card.pdf

[7] Z. Sheng, Z. Chen, S. Gu, H. Huang, G. Gu, and J. Huang, "LLMs in software security: A survey of vulnerability detection techniques and insights," ACM Computing Surveys, vol. 58, no. 5, pp. 1–35, 2025.

[8] Z. Pan, J. Liu, Y. Dai, and W. Fan, "Large language model-enabled vulnerability investigation: A review," in Proc. 2024 Int. Conf. Intelligent Computing and Next Generation Networks (ICNGN), pp. 1–5, 2024.

[9] F. Aguilera-Martínez and F. Berzal, "LLM security: Vulnerabilities, attacks, defenses, and countermeasures," arXiv preprint arXiv:2505.01177, 2025.

[10] J. Bommarito et al., "GPT as knowledge worker: A zero-shot evaluation of (AI) CPA capabilities," arXiv preprint arXiv:2301.04408, 2023.

[11] M. Bommarito II and D. M. Katz, "GPT takes the bar exam," arXiv preprint arXiv:2212.14402, 2022.

[12] E. Martínez, "Re-evaluating GPT-4's bar exam performance," Artificial Intelligence and Law, vol. 33, no. 3, pp. 581–604, 2025.

[13] B. S. Bloom, Taxonomy of Educational Objectives. New York, NY: Longman, 1956.

[14] T. Brown et al., “Language models are few-shot learners,” in Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020.

[15] K. Abramski et al., “The ‘LLM World of Words’ English free association norms generated by large language models,” Scientific Data, vol. 12, p. 803, 2025. doi: 10.1038/s41597-025-05156-9.

[16] Y. Wang et al., “The fluency-based semantic network of LLMs differs from humans,” Computers in Human Behavior: Artificial Humans, vol. 3, p. 100103, 2025.

[17] B. M. G. Nielsen, I. Macocco, and M. Baroni, “Prediction hubs are context-informed frequent tokens in LLMs,” in Proc. 63rd Annual Meeting of the Association for Computational Linguistics (ACL), pp. 23715–23745, 2025.

[18] E. M. Bender and A. Koller, “Climbing towards NLU: On meaning, form, and understanding in the age of data,” in Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 5185–5198, 2020.