

Automated Evaluation of Mathematical Equivalence Between Personalized and Standard Word Problems

Burcu Arslan¹, Ikkyu Choi¹, Jesse R. Sparks¹, Reginald M. Gooch¹,
Candace Walkington², Matthew L. Bernacki³

¹ETS Research Institute, 660 Rosedale Road, Princeton, NJ 08541

²Southern Methodist University, 6425 Boaz Lane, Dallas, TX 75205

³University of North Carolina at Chapel Hill, 145 E. Cameron Street, Chapel Hill, NC 27514

Correspondence: barslan@ets.org

Abstract

Generative AI enables real-time and scalable context personalization of mathematics word problems (MWP) based on students' self-reported interests during assessment. However, a question arises: are personalized and standard MWPs mathematically equivalent? In this paper, we present two Natural Language Processing pipelines for evaluating mathematical equivalence between personalized and standard MWPs.

1 Personalization of Math Word Problems

Traditional K–12 classroom mathematics assessments often rely on standardized, one-size-fits-all problem contexts. Although standardization supports data and construct comparability, math word problem (MWP) contexts may not be equally familiar or relevant to all students, which can limit motivation, engagement, and performance (Arslan et al., 2024; Bennett, 2024; Bernacki and Walkington, 2018; Sireci, 2020; Walkington, 2013). This is consequential because low motivation during assessment can reduce cognitive engagement and lead to performance that may not reflect students' true knowledge, skills, and abilities (Finn, 2015). Such disengagement can threaten the validity of score interpretations, since inferences about students' knowledge, skills, and abilities are based on the assumption that students put forth their best effort during the assessment (Wise, 2017; Arslan and Finn, 2023).

Scalable personalization offers a potential means of making assessment contexts more meaningful to individual students while maintaining comparability across examinees. Building on this premise, we developed Context-AI, a learner-driven, generative AI-enabled research prototype for classroom mathematics assessment that allows eighth-grade students to personalize problem contexts based

on their out-of-school interests. Context-AI aims to adapt MWPs developed by subject-matter experts to students' self-reported interests in real time, while preserving mathematical equivalence through isomorphic problem structures (Figure 1).

a) A Subject-Matter Expert Written Standard Word Problem Anton has some money saved for school. To increase the amount he has saved, he got a job and added all the money earned to the amount already saved. • Anton was paid a constant rate in dollars per hour at his job. • After working 10 hours, the amount he had saved was \$450. • After working 20 hours, the amount he had saved was \$600. • Anton did not save any other money or spend any of the money he had saved. Which equation represents y, the amount of money in dollars Anton would have saved after working x hours? Select one answer. ☐ $y = 10x + 150$ ☐ $y = 15x + 100$ ☐ $y = 15x + 300$ ☐ $y = 30x + 150$
b) A Personalized Word Problem Taylor creates TikTok videos for her channel. To increase her number of followers, she posted videos more frequently and added all the new followers earned to the numbers she already had. • Taylor gained followers at a constant rate in followers per hour. • After posting for 10 hours, she had 450 followers. • After posting for 20 hours, she had 600 followers. • Taylor did not gain or lose any other followers. Which equation represents y, the number of followers Taylor would have after posting x hours? Select one answer. ☐ $y = 10x + 150$ ☐ $y = 15x + 100$ ☐ $y = 15x + 300$ ☐ $y = 30x + 150$

Figure 1: a) An example standard MWP developed by a subject-matter expert, and b) an example of a personalized MWP (isomorphic to the standard problem) created through Context-AI based on the interest “Creating TikTok Videos.”

Within the scope of this paper, as an operational definition of assessment-level mathematical equivalence, a standard and a personalized MWP are considered mathematically equivalent when they contain the same numerical values and can be solved

using the same mathematical knowledge, operations, reasoning steps, and solution structure, despite differences in surface wording. Any change that alters the numerical values, mathematical demands, or solution process is treated as introducing non-equivalence. This conceptualization reflects a form of “flexible standardization,” in which comparability is achieved through equivalent measurement opportunities rather than identical items (Arslan et al., 2026). By presenting mathematically equivalent problems in contexts that are more familiar and personally meaningful to students, Context-AI is intended to reduce construct-irrelevant variance (Messick, 1995) associated with disengagement or contextual unfamiliarity, thereby enabling assessment scores to more accurately reflect students’ mathematical understanding.

Previous research on personalized learning shows that connecting mathematics with students’ interests enhances sense of belonging, intrinsic motivation, engagement, and achievement (Bernacki and Walkington, 2018; Norberg et al., 2025; Walkington, 2013), with medium-to-large effects on motivation and retention (Lin et al., 2024). Context-AI builds on this evidence by enabling real-time, learner-driven context personalization during classroom mathematics assessment.

Generative AI enables real-time scalable personalization of MWP based on students’ self-reported interests, but it also raises a central question for construct validity and comparability: Are personalized and standard MWPs mathematically equivalent?

We investigated this question using batch-generated personalized MWPs based on student interests collected through an online survey shared with schools. Although humans can evaluate equivalence, manual review is time consuming for large item pools and infeasible in real-time settings. Therefore, we developed two Natural Language Processing (NLP) pipelines for evaluating mathematical equivalence (a rule-based pipeline and an LLM-based pipeline), each with different advantages and disadvantages, and compared their performance with human ratings. The rule-based NLP pipeline provides reliable symbolic representations for predefined templates, but is less robust to linguistic variation and novel personalized MWP contexts. In contrast, the LLM-based NLP pipeline handles diverse natural languages more flexibly but may produce inconsistent judgments or hallucinations. Cases in which the two pipelines disagreed were treated as uncertain and reviewed by

humans to inform selection of the best-performing pipeline. Subsequently, a sample of the generated MWP dataset was then independently reviewed by humans to evaluate human-machine agreement with the best-performing pipeline. Because both pipelines use computational graphs, structured symbolic representations of the underlying mathematical operations, we next provide a brief overview of the previous work on using computational graphs for automated MWP solving before presenting the details of our methodology.

1.1 Computational Graphs for Automated Math Word Problem Solving

Automatically solving real-world tasks has been a long-standing goal of intelligent systems (Bobrow, 1964; Turing, 1950; Wang et al., 2025). Within this broader effort, a substantial body of research has focused specifically on automated methods for solving MWPs (Wang et al., 2025). These approaches aim to transform natural language problem statements into mathematical representations that can be solved computationally. To support this transformation, researchers have used a range of NLP methods, including neural networks and computational graphs.

For example, Zhang et al. (2020) introduced Graph2Tree, which analyzes the grammatical structure of a MWP to build two types of graphs: one that links each number to its descriptive context words and another that captures the numerical ordering of relationships between quantities. These graphs are then processed by a neural network to generate the correct solution expression. Graph2Tree constructs graphs as input representations for a trained neural network that generates solution expressions, using Stanford CoreNLP (Manning et al., 2014) and learning quantity representations through convolutional neural networks.

More recently, Wang et al. (2025) proposed KnBERT-TD, combining a knowledge-enhanced pre-trained language model with a binary expression tree decoder, demonstrating that richer semantic representations yield more accurate equation generation. KnBERT-TD uses a pre-trained language model to interpret a MWP and produce a structured symbolic representation. This representation is a binary expression tree that is used to generate a solution equation. KnBERT-TD relies on a deterministic computation step after symbolic expression generation, and uses this step to compute the answer.

Yang et al. (2024) used computational graphs for a different purpose: as structural signatures for retrieving analogous MWPs in few-shot prompting. The key insight is that two problems sharing the same computational graph (e.g., $x = \text{num_a} * \text{num_b} / \text{num_c}$) require the same sequence of arithmetic operations and therefore follow the same reasoning path, making them useful as worked examples to guide a general-purpose LLM in solving a new, structurally similar problem.

Overall, unlike previous work, we use computational graphs for an evaluation of mathematical equivalence in LLM-enabled context personalization. Before describing how computational graphs are incorporated into our rule-based and LLM-based NLP pipelines, we first detail the collection and processing of student interest data and the batch generation of personalized MWPs.

2 Methods

2.1 Student Interest Survey Data

To generate a batch of personalized MWPs, we collected data from 596 eighth-grade students (52.3% male, 44.3% female, 3.4% prefer to self-describe; 67.3% White, 11.7% Black or African American, 9.6% Asian or Asian American, 7.7% Hispanic or Latino, 3.7% American Indian or Alaskan Native) from seven schools and districts from four states in the U.S. (GA, NC, NJ, and PA). The online survey asked students to enter five of their out-of-school interests, resulting in 2,979 interests (due to one missing data point from one student).

Student interest data was processed through a two-stage process. *Stage 1 (text preprocessing and semantic deduplication)* transformed raw survey responses into a clean, semantically deduplicated flat file suitable for manual categorization of student interests by a human. *Stage 2 (category-constrained interest selection)* operated on the output with interest categories from Stage 1 to identify a diagnostically diverse set of interest exemplars by selecting interest from each category for use in batch personalized MWP generation (see Appendix A for details of these two stages).

2.2 Batch Generation of Personalized MWPs

During batch generation of personalized MWPs, we evaluated five LLMs accessed through Microsoft Azure APIs to identify a suitable model for the classroom use case of Context-AI: a) Claude Opus 4.1, b) DeepSeek V3-0324, c) Grok 3, d) GPT

5.1 Chat, and e) Llama 4 Maverick. For model-specific parameters, such as temperature, we used each model’s default parameter.¹

Personalized MWPs were generated based on a base set of five standard algebra MWPs for eighth-grade students selected from large-scale assessment contexts used domestically (see Appendix B). For each combination of selected interest, each LLM, and a base standard MWP (i.e., problem type), five personalized MWPs (i.e., repetitions) were produced. This process yielded a total of 3,327 personalized MWPs (after filtering out potentially inappropriate content with a safety guardrail we developed), which allowed us to examine both consistency and variation within and between models.

2.3 Automated Evaluation of Mathematical Equivalence Between Personalized and Standard (Base) MWPs

2.3.1 Rule-Based NLP Pipeline

The rule-based NLP pipeline (Figure 2) parses MWPs into computational graphs using linguistic dependency parsing (spaCy) and pattern matching, with no neural components involved in graph construction. Equivalence is then verified through a three-strategy cascade. For the "symbolic scaling" base problem type, (see Figure B2 in Appendix B), the derived graph strings are compared after replacing each placeholder name with a generic token, so that `mul(5, BOX)` and `mul(5, ROOM_ICON)` are correctly recognized as the same structure regardless of what the placeholder is called. For the remaining four base problem types, equivalence is checked by comparing the slope and intercept values extracted by each parser: for "two-point linear" and "rate-plus-fixed-fee" problem types both values can be non-zero (see Figures B1 and B3 in Appendix B), while for "proportional-table" and "rate-from-total" problems the intercept is always zero by definition (the relationship is purely proportional; see Figures B4 and B5 in Appendix B). Both values are compared using exact symbolic arithmetic via SymPy. Third, if neither of the first two strategies applies, the full derived algebraic expressions are compared symbolically via SymPy as a fallback.

¹Note that the goal of this study was not to compare LLMs, but to introduce an evaluation pipeline for on-the-fly context personalization using an LLM. Therefore, the use of multiple LLMs was secondary to the main purpose of this study. We evaluated different models only to identify the specific use case of context personalization of MWPs, rather than to make broad claims about their relative performance in general.

More precisely, to identify the mathematical structure of each problem, the pipeline tries five template parsers in a fixed order (i.e., two-point linear, rate-from-total, rate-plus-fee, symbolic scaling, and proportional table) and stops at the first one that succeeds. The order matters: the most selective parsers run first. Each early parser checks for specific structural features and actively rejects problems that belong to a later template. For example, the "two-point linear" parser screens out problems with a fixed fee or equal-distribution language before they can be misclassified. This ensures that narrow, well-defined problem types are identified before broader ones, keeping classification errors low across all five categories. For each template, the pipeline produces two graph components: a constraint graph, which encodes the full system of equations as an abstract syntax tree (AST; e.g., `system(equal(Total, add(mul(48, h), 9))))`, and a derived expression graph, which isolates the simplified algebraic expression that directly answers the problem question (e.g., `add(mul(48, h), 9)`). The derived expression is then passed to SymPy for the verification of deterministic algebraic equivalence against the corresponding base item.

2.3.2 LLM-Based NLP Pipeline

Unlike the rule-based pipeline, the LLM-based pipeline (Figure 3) uses an LLM to convert an MWP into a structured mathematical representation. The LLM-based pipeline combines LLM inference with deterministic computer algebra (i.e., a neuro-symbolic approach) to assess whether LLM-generated MWPs are mathematically equivalent to base standard MWPs. The approach assigns the linguistically complex task of word-problem interpretation to GPT 5.2 Chat², accessed via the Azure OpenAI API, while delegating algebraic verification to a symbolic algebra engine.

More specifically, each MWP text was submitted to the LLM via a two-part message. The system message established the role of the model as "an expert mathematical NLP extractor." The user message instructed the model to analyze the problem text and extract its computational graph, following three guidelines: (a) identify the independent and dependent variables, (b) express the derived algebraic relationship as a SymPy-parseable string

using strict Python math notation (e.g., using `*` for multiplication), and (c) return `None` if the text did not contain a solvable algebraic problem or lacked sufficient numerical data.

Crucially, the LLM response was not free-form text. It was constrained to a predefined structured output schema (via OpenAI's Pydantic structured outputs API), which required the model to return exactly seven fields: a Boolean flag indicating whether the text was a math problem; a template type classification ("two-point linear", "rate from total", "rate plus fixed fee", "symbolic scaling", "proportional table", or "unknown"); the independent and dependent variable names; a constraint graph expressed as an AST; a simplified derived graph AST; and the derived expression as a SymPy-parseable algebraic string. This schema enforcement ensured that the LLM output was machine-readable and directly usable by the symbolic verification step, with no additional parsing required. If the LLM call failed, it was retried up to three times with a two-second backoff before the item was marked as a failed extraction. Note that the template type was inferred by the LLM from the item text alone. However, for consistency with the rule-based pipeline, the LLM-inferred labels were automatically verified and, where necessary, corrected via a post-hoc override table. This correction only determined how the extracted expressions were cleaned up before comparison (e.g., whether to remove the left-hand side of an equation or whether to treat different variable names as equivalent); the derived algebraic expression itself remained entirely LLM-generated. The rule-based pipeline, by contrast, performed both template classification and expression derivation automatically without any correction. Both rule-based and LLM-based NLP pipelines generated output with values of "True" (i.e., Equivalent), "False" (Non-Equivalent), or NA. The first author reviewed the NA cases and classified them as "False" (Non-Equivalent), since the personalized MWPs in this category were not equivalent to their base standard MWP counterparts.

2.4 Evaluation of Automated Pipelines with Human Raters

There were two stages of human review. The goal of the first stage was to identify the pipeline that was more in agreement with human judgment among cases in which the two pipelines disagreed. To this end, the mathematical equivalence of 3,327

²At the time of batch personalized MWP generation, GPT 5.2 Chat was not yet available; therefore, we used GPT 5.1 Chat for batch generation and GPT 5.2 Chat for the later-developed LLM-based NLP pipeline.

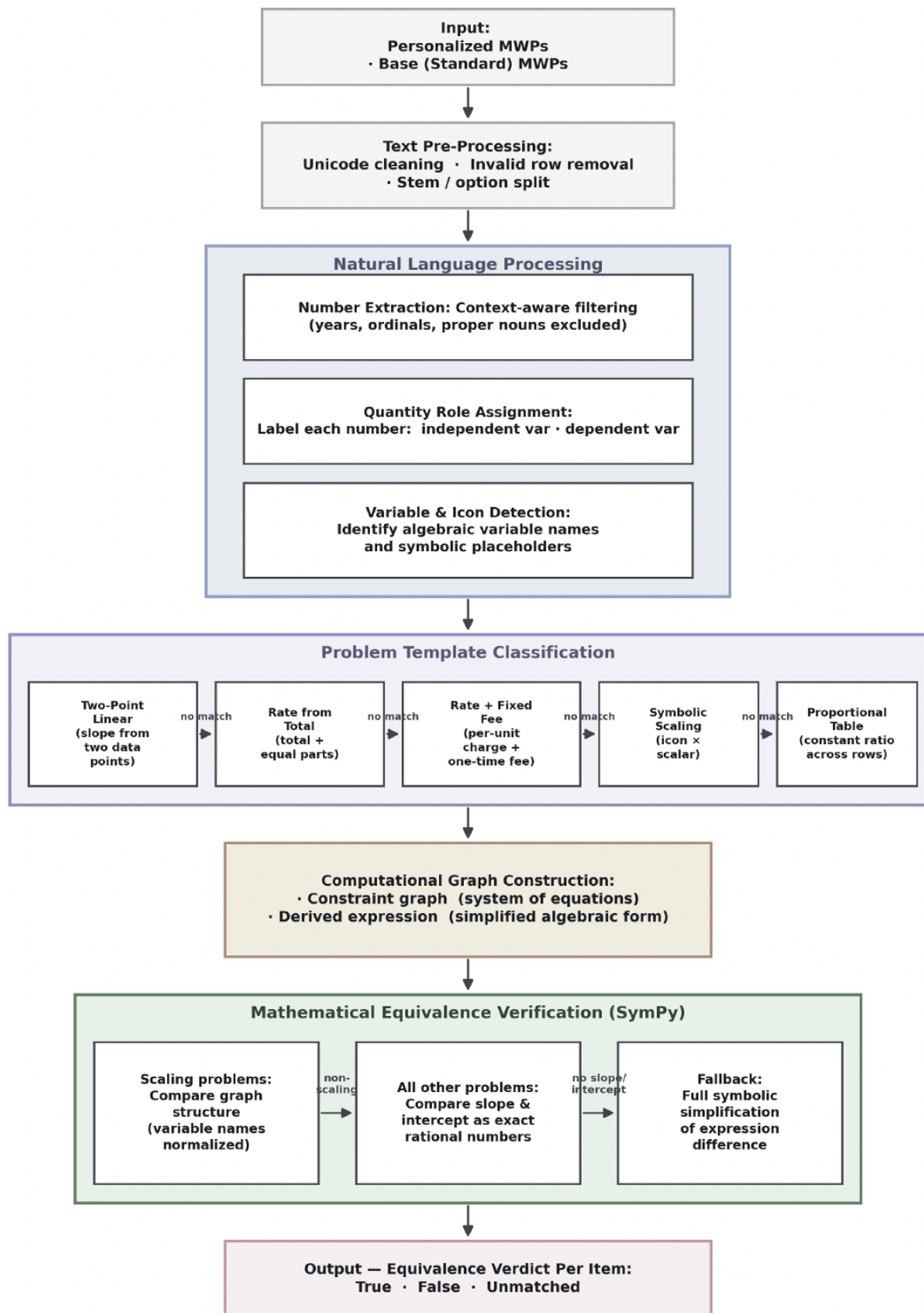


Figure 2: Rule-based NLP architecture for mathematical equivalence. "True" indicates "Equivalence" and "False" and "unmatched" indicate "Non-Equivalence".

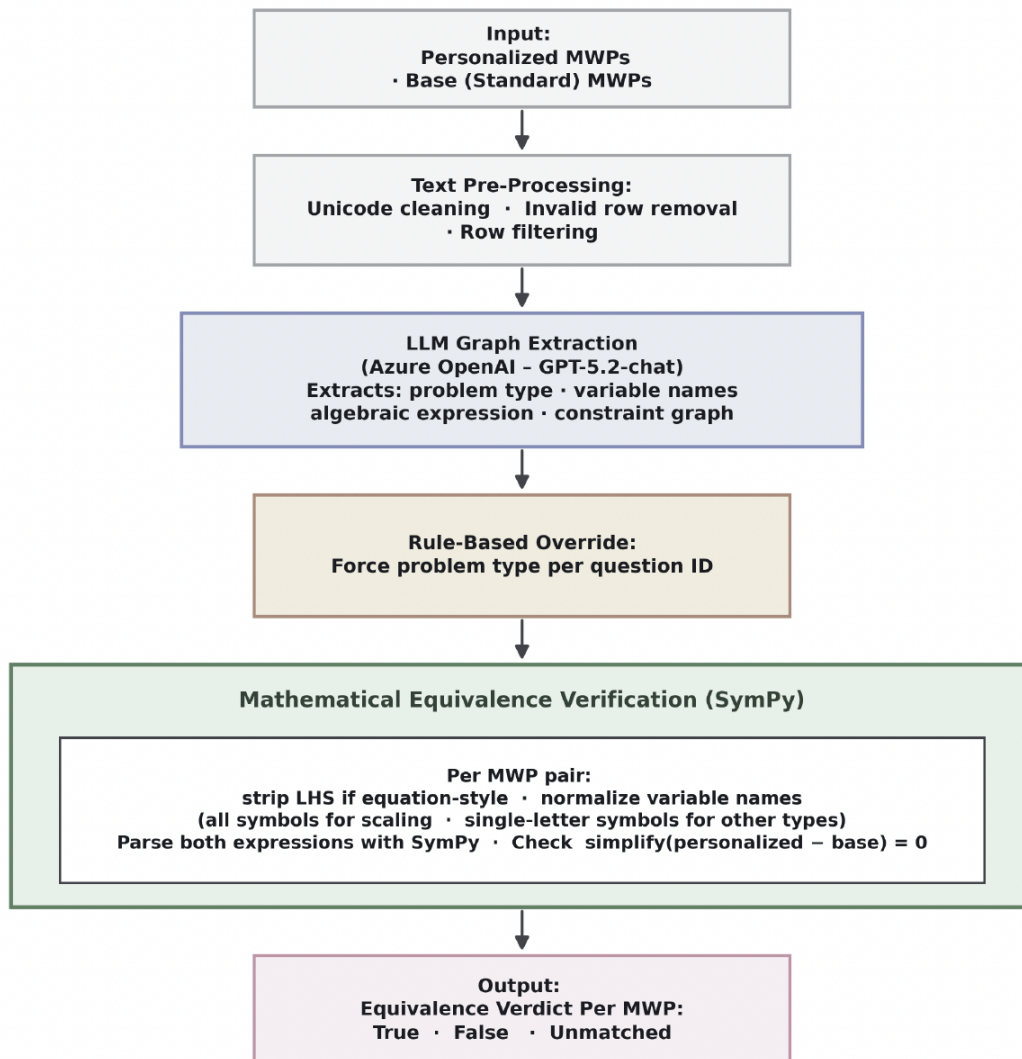


Figure 3: LLM-based NLP architecture for mathematical equivalence. "True" indicates "Equivalence" and "False" and "unmatched" indicate "Non-Equivalence".

personalized MWP generated during batch item generation was compared using rule-based and LLM-based NLP pipelines and their equivalence verdicts were compared. The cases in which the two pipelines disagreed were independently reviewed by the first and fourth authors, who adjudicated which verdict better matched human judgment. This calibration exercise served as a pipeline selection step: the pipeline that showed greater agreement with human reviewers on disputed cases was selected for the full human validation study in Stage 2. This selection was a practical necessity — conducting a full stratified human review of 600 items for both pipelines simultaneously was beyond available annotation resources — rather than a claim that the other pipeline has no merit.

The goal of the second stage was to investigate the human-machine agreement for the identified pipeline based on the results of the first stage. To do this, we conducted a stratified human review of personalized MWPs from the final data set of 3,327 personalized MWPs. Because "False" (Non-Equivalent) cases were relatively rare and diagnostically important, all 134 non-equivalent MWPs are included in the human-review sample. We then sampled 466 equivalent MWPs using stratified sampling across LLM models and base standard MWPs, yielding a main validation sample of 600 personalized MWPs for human review (18% of the total sample). Horvitz-Thompson (HT) design weights were assigned to each sampled MWP to allow population-level inference (Horvitz and Thompson, 1952). All automated non-equivalent MWPs received a weight of 1. For equivalent MWPs, a stratum-specific weight was assigned to each item as ($w_i = N_{\text{stratum}}/n_{\text{stratum}}$), where N_{stratum} is the total number of equivalent MWPs in a given *Model* $\times$ *base-problem* stratum and n_{stratum} is the number sampled from that stratum. Because sampling was stratified rather than a single simple random draw, weights varied across strata and were not a single global ratio. Confidence intervals at the 95% level were estimated using a non-parametric stratified bootstrap ($B = 2,000$ resamples), resampling independently within the automated non-equivalent stratum (census) and the automated equivalent stratum (stratified sample), preserving the original design structure.

The first and fourth authors independently and without the knowledge of the model identity and automated equivalence labels judged whether each generated MWP is mathematically equivalent to its

corresponding base standard MWP using a guideline which gives a definition of mathematical equivalence and provided worked examples of equivalent and non-equivalent cases together with a decision guide for borderline cases (e.g., same numbers, same math operations and steps $\rightarrow$ Equivalent; one problem requires an extra operation $\rightarrow$ Non-Equivalent). Disagreements were resolved through adjudication. Pre-adjudication human-human reliability was evaluated using percent agreement, Cohen’s κ , and Gwet’s AC1. Machine-human agreement was evaluated by comparing the equivalence pipeline’s labels that was identified at Stage 1 of human review against adjudicated human reference labels at Stage 2 of human review using the weighted confusion matrix, sensitivity, specificity, balanced accuracy, Cohen’s κ , Gwet’s AC1, and 95% CIs. Note that the discrepancy between Cohen’s κ , and Gwet’s AC1 is expected given the low prevalence of non-equivalent items: Cohen’s κ is known to be deflated when one class is rare (the prevalence-bias problem), whereas Gwet’s AC1 is more robust to class imbalance. Together, these metrics cover correctness, error decomposition, imbalance robustness, and agreement reliability without redundancy.

3 Results

3.1 Mathematical Equivalence: Comparison of Rule-Based and LLM-Based NLP Pipelines

Figure 4 shows the mathematical equivalence pass rate (i.e., the percentage of personalized MWPs judged as mathematically equivalent to their base counterparts) of the rule-based and LLM-based NLP pipelines. As can be seen in the figure, there are some discrepancies between the pass rates of the two pipelines. In terms of LLM models, both pipeline results indicated that, in our current design of the study, GPT 5.1 Chat, Claude Opus 4.1, and Grok 3 models created more mathematically equivalent personalized MWPs compared to DeepSeek V3-0324 and Llama 4 Maverick models across problem types. The results showed that in 74 out of 3,327 cases ($\approx 2\%$), rule-based and LLM-based NLP pipelines returned problems that did not match their predictions. The first and fourth authors independently reviewed these discrepant cases in which two pipelines did not agree whether personalized MWPs match their base counterparts (for the confusion matrix, see Table 1). The human raters showed 86.5% pre-adjudicated agreement,

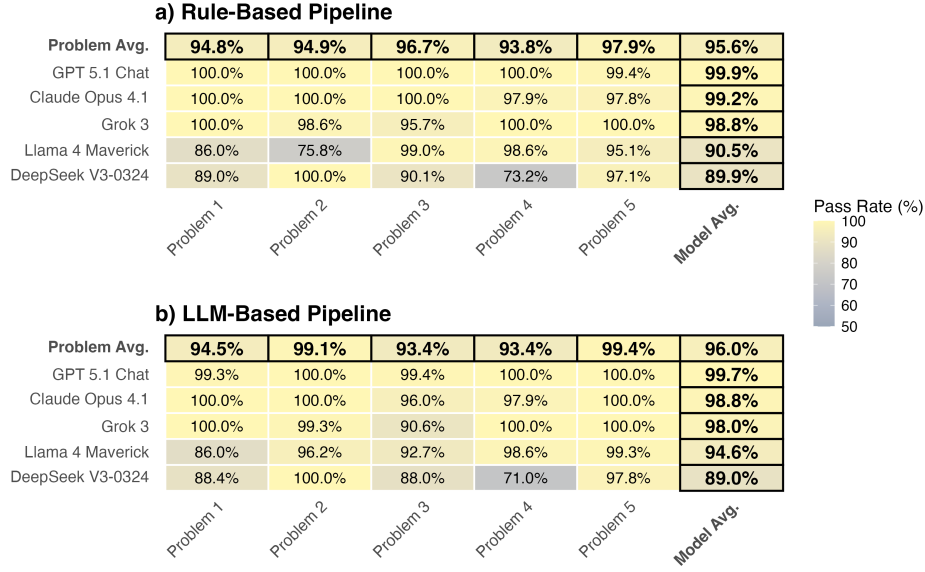


Figure 4: Mathematical equivalence pass rates for personalized MWPs across (a) the rule-based pipeline and (b) the LLM-based pipeline, disaggregated by the LLM used to generate the personalized problems and by problem type (see Appendix B). Pass rate is the percentage of personalized MWPs judged mathematically equivalent to their corresponding base MWPs.

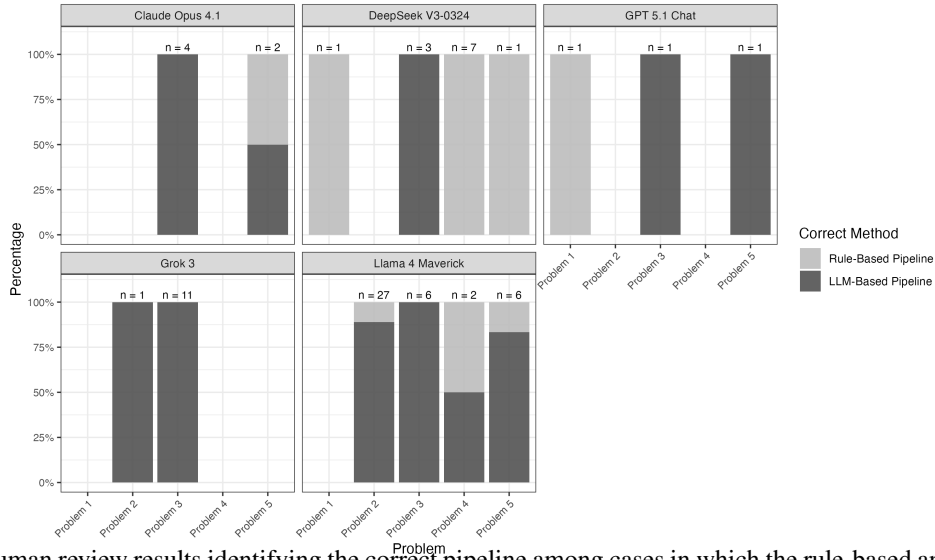


Figure 5: Human review results identifying the correct pipeline among cases in which the rule-based and LLM-based pipelines disagreed, disaggregated by the LLM used to generate the personalized problems and by problem type.

with Cohen’s $\kappa = .62$, and Gwet’s $AC1 = .79$.

Human Reviewer 1	Human Reviewer 2		
	Rule-Based Correct		LLM-Based Correct
	Rule-Based Correct	LLM-Based Correct	Correct
	13 (17.56%)	5 (6.76%)	51 (68.92%)

Table 1: Confusion matrix comparing Reviewer 1 and Reviewer 2 judgments of which pipeline made the correct decision among the non-matching cases.

After resolving the disagreed cases, we found that the LLM-based pipeline made the correct decision in 57 out of 74 cases (77.03%; 95% CI [65.76%, 86.01%]). An exact binomial test showed that this proportion was significantly higher than the chance-level performance of 50%, $p < .001$, in-

dicating that the LLM-based pipeline was more often agreed with humans than the rule-based pipeline among the cases where the two methods disagreed (see Figure 5). Therefore, we decided to move forward with the LLM-based pipeline to evaluate the human-machine agreement for mathematical equivalence of personalized MWPs with their base standard counterparts.

3.2 Comparison of the Agreement Between Human and the LLM-Based NLP Pipeline

The pre-adjudication results of a main validation sample of 600 personalized MWPs (18% of the total sample) indicated that the human raters showed

94.7% agreement, with Cohen’s $\kappa = .86$. Gwet’s AC1, which is robust to the paradoxes of Cohen’s κ under skewed category prevalence, was .92 (SE = .02, 95% CI [.885, .945], $p < .001$).

Table 2 presents the weighted confusion matrix comparing the LLM-based NLP pipeline labels against the adjudicated human rater labels. The confusion matrix is weighted because each evaluated case contributes according to its sampling weight rather than as a single unweighted observation. Therefore, the cell values should be interpreted as weighted counts, not as raw numbers of items.

		Human	
		Non-eq.	Eq.
LLM-based pipeline	Non-eq.	127.00 (3.82%)	7.00 (0.21%)
	Eq.	177.84 (5.35%)	3015.16 (90.62%)

Table 2: Weighted confusion matrix comparing LLM-based pipeline labels with human-adjudicated labels. Values are weighted counts, with percentages of the total weighted sample in parentheses.

Non-equivalent ("False") items are treated as the positive class (i.e., the category the pipeline is designed to flag). This choice reflects both the pipeline’s purpose — catching mathematically flawed items — and the rarity of non-equivalent cases ($\approx 4\%$ of the sample), where accuracy on the majority class would otherwise obscure detection failures. Because non-equivalent is treated as the positive class, 127.00 (3.82%) represents weighted true positives, where both the LLM-based pipeline and human raters labeled items as non-equivalent. The value 7.00 (0.21%) represents weighted false positives, where the LLM-based pipeline labeled items as non-equivalent but human raters labeled them as equivalent. In contrast, 177.84 (5.35%) represents weighted false negatives, where the LLM-based pipeline labeled items as equivalent but human raters labeled them as non-equivalent. Finally, 3015.16 (90.62%) represents weighted true negatives, where both the LLM-based pipeline and human raters labeled items as equivalent. Thus, the matrix indicates that the LLM-based pipeline produced few weighted false positives but missed a larger number of weighted non-equivalent cases.

As shown in Table 3, the LLM-based pipeline was very good at identifying equivalent items. Specificity was .998, meaning that items humans judged as equivalent were almost never incorrectly flagged as non-equivalent. Positive predictive value was also high (.948), meaning that when the pipeline flagged an item as non-equivalent, human

Metric	Estimate	95% CI
Specificity	.998	[.996, .999]
Positive Predictive Value	.948	[.910, .978]
Sensitivity	.417	[.340, .523]
Negative Predictive Value	.944	[.923, .964]
Accuracy	.944	[.924, .963]
Balanced accuracy	.707	[.669, .761]
Cohen’s κ	.554	[.467, .657]
Gwet’s AC1	.937	[.911, .959]

Table 3: LLM-based pipeline performance against human-adjudicated labels.

raters usually agreed. However, sensitivity was lower (.417). This means the pipeline detected about 42% of the non-equivalent items identified by human raters and missed about 58% of them. Negative predictive value was high (.944), meaning that items labeled as equivalent by the pipeline were usually confirmed as equivalent by human raters. Overall accuracy was high (.944), but balanced accuracy was lower (.707), showing that performance was uneven across equivalent and non-equivalent items. Agreement was high according to Gwet’s AC1 (.937), but lower according to Cohen’s κ (.554). As we mentioned earlier, this difference is expected because Cohen’s κ is affected by class imbalance; in this study, most cases were equivalent, which lowered κ even though overall agreement was high.

4 Conclusions

This study presented two NLP pipelines to evaluate whether LLM-enabled context personalization of math word problems (MWP) preserves mathematical equivalence with their corresponding base MWPs. Across 3,327 personalized MWPs generated, the rule-based and LLM-based pipelines showed broadly similar patterns, while also revealing some discrepancies that required human review. For the non-matching cases between the rule-based and the LLM-based pipelines, the LLM-based pipeline agreed more often with human judgments and was therefore selected for further human-machine validation (see Table 1 and Figure 5).

The human-machine validation results indicate that the LLM-based pipeline can be characterized as a high-precision, low-recall detector: its positive flags are highly trustworthy, but it should not be relied upon to identify all non-equivalent items in an unreviewed dataset (see Table 3).

An initial qualitative examination of all false-negative cases (i.e., MWPs the pipeline classified as equivalent but human raters adjudicated as non-equivalent) suggests that one important fail-

Example False Negative: Same Equation, Missing Narrative Grounding

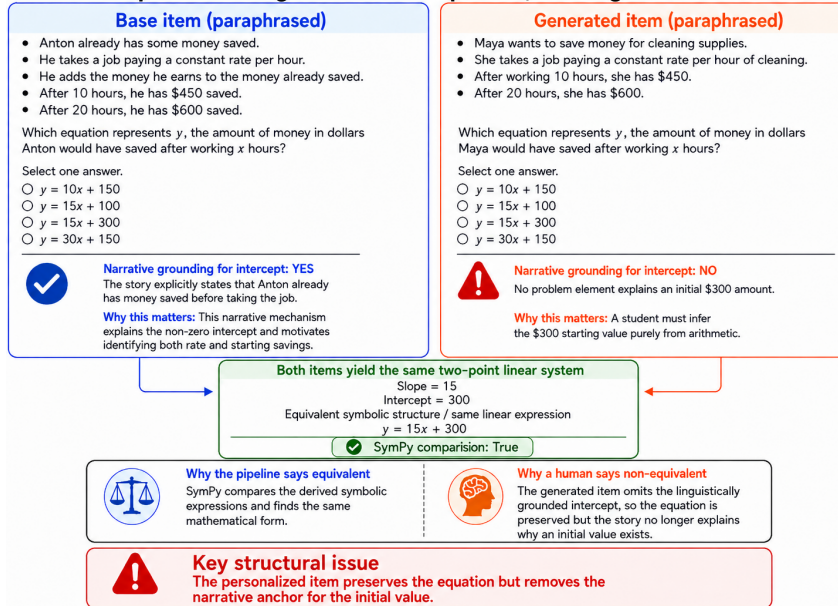


Figure 6: Example of a false negative caused by missing narrative grounding: the LLM-based pipeline classified the personalized MWP as mathematically equivalent to its base counterpart, whereas human raters adjudicated it as non-equivalent.

ure mode is not algebraic reasoning but linguistic grounding. Figure 6 shows an example of a false negative case with missing narrative grounding. The LLM-based NLP pipeline extracts a symbolic expression from each problem text through an LLM-structured extraction step and then uses SymPy to verify the symbolic equivalence between the base and generated expressions. This design is effective in detecting algebraic mismatches, but it can miss cases in which two problems yield the same expression.

5 Limitations and Future Directions

This study has several limitations. First, the rule-based pipeline depends on predefined templates for the five types of personalized MWPs used in stress testing and, therefore, cannot generalize to problems outside its linguistic or structural coverage. Future research should expand these templates. Second, while the LLM-based pipeline is more flexible and can potentially be used beyond the five templates, it is structurally limited in capturing semantic constraints essential for determining whether two problems are mathematically equivalent. Human raters flagged missing narrative grounding because the rubric required preservation of not only the same symbolic expression but also the quantities, relationships, operations, solution path, and contextual rationale that make the expression meaningful. Future work could incorporate this rubric into the LLM evaluation stage through

rubric-guided prompting, few-shot examples from human-adjudicated cases, or retrieval from a corpus of confirmed equivalent and non-equivalent MWPs. Third, although the human review sample for the LLM-based pipeline was designed to be population-representative through Horvitz-Thompson weighting, it included only 600 items from a larger item pool.

Overall, the findings position the LLM-based pipeline as a promising but still improvable tool for confirming items flagged as mathematically non-equivalent. However, items flagged as equivalent still require human sampling or improved detection methods to support broader quality monitoring.

Acknowledgments

The authors thank Thomas Florek for implementing the batch item generation software and the Context-AI research prototype.

References

- Burcu Arslan, Randy E. Bennett, Sandip Sinharay, Jesse R. Sparks, and Kadriye Ercikan. 2026. Personalization in assessment to optimize engagement and performance. In Xinhui Xiong, Mark D. Shermis, and Jiawei Xiong, editors, *The Role of AI in Assessment: Revolutionizing Education*. Taylor & Francis/Routledge.
- Burcu Arslan and Bridgid Finn. 2023. The effects of personalized nudges on cognitively disengaged student behavior in low-stakes assessments. *Journal of Intelligence*, 11(11):204.

- Burcu Arslan, Blair Lehman, Caitlin Tenison, Jesse R. Sparks, Alexis A. López, Lin Gu, and Diego Zapata-Rivera. 2024. Opportunities and challenges of using generative AI to personalize educational assessment. *Frontiers in Artificial Intelligence*, 7:1460651.
- Max Bachmann, layday, thomasryde, Georgia Kokkinou, Henry Schreiner, Jeppe Fihl-Pearson, dheeraj, Vioshim, pekkarr, Michał Górny, Moshe Sherman, Cristian Le, odidev, glynn, Trenton H., tr halfspace, dotlambda, Nicolas Renkamp, Kwuang Tang, and 6 others. 2025. [rapidfuzz/rapidfuzz: Release 3.13.0](#).
- Randy E. Bennett. 2024. Personalizing assessment: Dream or nightmare? *Educational Measurement: Issues and Practice*, 43(4):119–125.
- Matthew L. Bernacki and Candace Walkington. 2018. The role of situational interest in personalized learning. *Journal of Educational Psychology*, 110(6):864.
- Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc., Sebastopol, CA.
- Daniel Bobrow. 1964. *Natural language input for a computer problem solving system*. Ph.D. thesis, MIT, Boston, MA.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [BERT: pre-training of deep bidirectional transformers for language understanding](#). *CoRR*, abs/1810.04805.
- Christiane Fellbaum. 1998. [WordNet: An Electronic Lexical Database](#). Language, Speech and Communication. Mit Press.
- Bridgid Finn. 2015. Measuring motivation in low-stakes assessments. *ETS Research Report Series*, 2015(2):1–17.
- Daniel G. Horvitz and Donovan J. Thompson. 1952. [A generalization of sampling without replacement from a finite universe](#). *Journal of the American Statistical Association*, 47(260):663–685.
- Lijia Lin, Xin Lin, Xiaofang Zhang, and Paul Ginns. 2024. The personalized learning by interest effect on interest, cognitive load, retention, and transfer: A meta-analysis. *Educational Psychology Review*, 36(3):88.
- Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. [The Stanford CoreNLP natural language processing toolkit](#). In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.
- Samuel Messick. 1995. Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *American Psychologist*, 50(9):741.
- David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguistic Investigations*, 30(1):3–26.
- Kole Norberg, April Murphy, Logan De Ley, Ethan Shafran Moltz, Husni Almoubayyed, and Steve Ritter. 2025. Using generative AI to foster student sense of belonging in mathematics. In *International Conference on Artificial Intelligence in Education*, pages 188–195. Springer.
- Stephen G. Sireci. 2020. Standardization and understandardization in educational assessment. *Educational Measurement: Issues and Practice*, 39(3):100–105.
- Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. [Mpnet: Masked and permuted pre-training for language understanding](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867. Curran Associates, Inc.
- Erik F. Tjong Kim Sang and Fien De Meulder. 2003. [Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition](#). In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.
- Alan M. Turing. 1950. Computing machinery and intelligence. *Mind*, pages 23–65.
- Candace Walkington. 2013. Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. *Journal of Educational Psychology*, 105(4):932.
- Yanli Wang, Ming Yan, Pengpeng Jian, Yangrui Yang, and Yang Li. 2025. A machine solution for math word problems based on semantic understanding enhancement. *Scientific Reports*, 15(1):36565.
- Zhiyuan Wang and Gade Pandu Rangaiah. 2017. Application and analysis of methods for selecting an optimal solution from the pareto-optimal front obtained by multiobjective optimization. *Industrial & Engineering Chemistry Research*, 56(2):560–574.
- Steven L. Wise. 2017. Rapid-guessing behavior: Its identification, interpretation, and implications. *Educational Measurement: Issues and Practice*, 36(4):52–61.
- Xiaocong Yang, Jiacheng Lin, Ziqi Wang, and Chengxiang Zhai. 2024. Learning by analogy: Enhancing few-shot prompting for math word problem solving with computational graph-based retrieval. *ArXiv Preprint ArXiv:2411.16454*.
- Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. [Graph-to-tree learning for solving math word problems](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937, Online. Association for Computational Linguistics.

A Appendix

A.1 Stage 1: Text Preprocessing and Semantic Deduplication

In stage 1, each response was processed through a sequential cleaning procedure. To standardize the text and prepare it for semantic representation, we applied a multi-step preprocessing NLP pipeline. Initially, all responses underwent Unicode normalization using Normalization Form Compatibility Composition (NFKC) and structural cleaning, including the extraction of specific terms from parentheticals (e.g., simplifying "Gaming (Fortnite)" to the more specific entity "Fortnite").

To ensure that relevant terms and brand names were not mistakenly treated as spelling or formatting errors during text cleaning, we used a three-step named-entity recognition (NER) strategy before normalizing the text (Nadeau and Sekine, 2007). This module identified proper names using: a) exact and fuzzy matching (via the rapidfuzz library; Bachmann et al., 2025) against a predefined, domain-specific lexicon of popular culture entities, such as Roblox, Fortnite, NBA, b) a transformer-based NER model (dslim/bert-base-NER; Devlin et al., 2018; Tjong Kim Sang and De Meulder, 2003) tuned for standard entity groups (Person, Organization, Location, Miscellaneous), such as "Drew Starkey", "Siege X", "MLB The Show 25", and c) a rule-based fallback using Natural Language Toolkit's (NLTK; Bird et al., 2009) part-of-speech tagger and chunker.

Identified proper names were temporarily masked while the remaining text was evaluated for structural noise. A heuristic noise score (range: 0.0 to 1.0³) was computed based on the ratio of non-alphanumeric and non-whitespace characters (weight = .40), four or more consecutive repeated characters (weight = .50), and digit ratios greater than 40% (weight = .10).

After noise scoring, proper names were temporarily replaced with underscore-joined placeholder tokens to preserve them through tokenization. The remaining alphabetic tokens were lemmatized using WordNet lemmatizer (Fellbaum, 1998) with part-of-speech informed lemma lookup (adjective, noun, verb, adverb), and stopwords were removed. Proper-name placeholders were then restored as underscore-joined lowercase strings. To-

³A score of 0.0 was assigned to responses composed entirely of identified proper names, as these contain no residual noise signal.

kens were required to be purely alphabetic or alphanumeric; other residual tokens were discarded.

Responses were retained for embedding and deduplication if they yielded at least one token, at least three characters, and no more than eight tokens after cleaning. The eight-token limit reflects that students usually give short interest survey responses. It also prevents long, sentence-like responses from affecting the analysis too much.

To group conceptually identical or highly similar responses, we employed a semantic deduplication strategy. Processed text sequences were embedded into 768-dimensional vector representations using a pre-trained sentence-transformer model (all-mpnet-base-v2; Song et al., 2020), which maps sentences and paragraphs to a shared semantic space.

We computed pairwise cosine similarities across all unique embeddings. A greedy clustering approach was applied to group responses that exhibited a cosine similarity score of ≥ 0.92 . From each group of near duplicates, a single canonical representative was selected. To ensure maximum clarity for downstream human coding, the representative was chosen by minimizing the previously computed heuristic noise score, using character length as a tie-breaker (favoring slightly longer, non-noisy formulations). The final corpus of deduplicated canonical interests was subsequently exported for manual interest category coding (870 unique interests).

The first author semantically categorized 870 unique interests, resulting in 31 unique interest categories (e.g., sports, music, traveling).

A.2 Stage 2: Category-Constrained Interest Selection

As input for stage 2, we used the file with the human-coded interest categories. All responses were re-encoded using the same all-mpnet-base-v2 model and normalization procedure as stage 1, ensuring that similarity scores in stage 2 were computed in the same metric space as the deduplication threshold applied in stage 1. Selections were made separately within each human-coded category for the representativeness of student interest input. If a category had fewer than three items after removing duplicates, all items in that category were kept and labeled as "Small Category". These categories were too small for a meaningful Pareto front-based selection approach⁴, which was used to identify

⁴Only the static front computation is used (no evolutionary operators).

interests that represented meaningful trade-offs between semantic representativeness and response quality (Wang and Rangaiah, 2017). This approach allowed us to select items that were not simply the most typical or the noisiest, but those that provided informative boundary cases for batch personalized MWP generation for stress-testing Context-AI.

For each category with at least three items, we calculated two scores for each item. The centrality scores measured how close an item was to the average meaning of the category. Items with higher centrality were more typical or representative of that category. The noise scores came from the stage 1 preprocessing step. A higher noise score means that the response was less clean or less standard, such as having unusual spelling, symbols, or formatting. Both scores were scaled to a range from 0 to 1.

We then treated centrality and noise as two separate goals. We wanted to identify items that were strong on one or both of these goals. To do this, we used a Pareto front approach (Wang and Rangaiah, 2017). Items on the Pareto front were those that could not be improved on one score without becoming worse on the other. In other words, they represented the trade-off between being representative of the category and being noisy or non-standard.

For each interest category, we selected items based on the following roles: a) *Central prototype*: The Pareto front item with the highest cosine similarity to the within-category centroid. This item was the most representative response for a given human-coded interest category; b) *Semantic outlier*: The Pareto front item with the lowest cosine similarity to the within-category centroid, provided that it differed from the central prototype. This item was the least typical response within the category and served as an edge case to evaluate how well the system handles peripheral interest inputs.

To make sure the same item did not appear under multiple categories, we used a global exclusion list. Once an item was selected for one category, it could not be selected again for another category. The interest categories were processed in alphabetical order, which is the default behavior in pandas. As a result, categories processed later had a slightly smaller pool of available items.

Because our goal was to stress test Context-AI, we selected interests from the Semantic Outlier and Small Category groups. We did not use the Central prototype items. This process produced 26 interests from 22 human-coded interest categories.

After the automated data-driven interest-selection process, the authors reviewed both the selected list and the full list. They removed four interests that were not specific enough for personalization and manually added 17 interests from student survey data and five inputs that were not in the student survey data but added for safety stress testing, which is outside of the scope of this paper. This process resulted in a final set of 44 interests (e.g., "Halloween/Christmas", "hip-hop dancing", "Pre-Cenozoic eras", "Car Racing", "MLB The Show 25", "my shows!!!", "extended-family events", including gibberish input "aaaaaaaaaaaaaaaa").

B Appendix

B.1 Problem 1: Two-Point Linear Type Base Math Word Problem

Anton has some money saved for school. To increase the amount he has saved, he got a job and added all the money earned to the amount already saved.

- Anton was paid a constant rate in dollars per hour at his job.
- After working 10 hours, the amount he had saved was \$450.
- After working 20 hours, the amount he had saved was \$600.
- Anton did not save any other money or spend any of the money he had saved.

Which equation represents y , the amount of money in dollars Anton would have saved after working x hours?

Select one answer.

- ☐ $y = 10x + 150$
- ☐ $y = 15x + 100$
- ☐ $y = 15x + 300$
- ☐ $y = 30x + 150$

Figure B1: Problem 1: Two-point linear type base MWP.

B.2 Problem 2: Symbolic Scaling Type Base Math Word Problem

If $\square$ represents the number of newspapers that Lee delivers each day, which of the following represents the total number of newspapers that Lee delivers in 5 days?

- A. $5 + \square$
- B. $5 \times \square$
- C. $\square + 5$
- D. $(\square + \square) \times 5$

Figure B2: Problem 2: Symbolic scaling type base MWP.

B.3 Problem 3: Rate Plus Fixed Fee Type Base Math Word Problem

A plumber charges customers \$48 for each hour worked plus an additional \$9 for travel. If h represents the number of hours worked, which of the following expressions could be used to calculate the plumber's total charge in dollars?

- A. $48 + 9 + h$
- B. $48 \times 9 \times h$
- C. $48 + (9 \times h)$
- D. $(48 \times 9) + h$
- E. $(48 \times h) + 9$

Figure B3: Problem 3: Rate plus fixed fee type base MWP.

B.4 Problem 4: Proportional Table Type Base Math Word Problem

Data	Mon.	Tues.	Wed.	Thurs.	Fri.	Sat.
Number Sold, n	4	0	5	2	3	6
Profit, p	\$2.00	\$0.00	\$2.50	\$1.00	\$1.50	\$3.00

Angela makes and sells special-occasion greeting cards. The table above shows the relationship between the number of cards sold and her profit. Based on the data in the table, which of the following equations shows how the number of cards sold and profit (in dollars) are related?

- A. $p = 2n$
- B. $p = 0.5n$
- C. $p = n - 2$
- D. $p = 6 - n$
- E. $p = n + 1$

Figure B4: Problem 4: Proportional table type base MWP.

B.5 Problem 5: Rate From Total Type Base Math Word Problem

Sara plans to exercise for 315 minutes over a period of 7 days.

She is going to exercise an equal number of minutes each day.

Which of the following equations represents the number of minutes, m , Sara has exercised after d days?

- A ☐ $m = 45d$ ☐
- B ☐ $m = 315d$ ☐
- C ☐ $m = 315 + d$ ☐
- D ☐ $m = \frac{315}{d}$ ☐
- E ☐ $m = \frac{45}{d}$ ☐

Clear Answer

Figure B5: Problem 5: Rate from total type base MWP.