11th International Scientific Conference
Technics, Informatics, and Education – TIE 2026
18-20 September 2026

Session: *Educational Technology*
Original scientific paper
DOI: https://doi.org/10.46793/TIE26.051D

Robustness Testing of Educational Chatbots Using Behavior-Driven Scenarios and Metamorphic Relations

Olga Dukić[1*] [0009-0004-1694-9592], Danijela Milošević[1] [0000-0002-4763-7012] and Jianxun Cui[2] [0000-0001-8286-490X]

[1] University of Kragujevac, Faculty of Technical Sciences Čačak, Čačak, Serbia
[2] Harbin Institute of Technology, School of Transportation Science and Engineering, Harbin, China
[*] dukicolga678@gmail.com

Abstract: *Artificial intelligence-supported educational chatbots are increasingly used to explain concepts, guide learners through tasks, support formative assessment, and recommend learning materials. Their reliability cannot be evaluated only through fixed dialog scenarios, because learners express the same educational need through paraphrases, incomplete messages, spelling errors, informal language, uncertainty, or direct requests for solutions. This paper proposes an extension of behavior-driven dialog testing with metamorphic relations for robust testing of educational chatbots. The approach defines controlled input transformations under which the chatbot should preserve or appropriately adapt its intended behavior. The proposed relations address paraphrasing, word-order variation, noisy input, formality variation, learner uncertainty, equivalent topic requests, requests for direct solutions, and repeated misunderstanding. Instead of requiring a single fixed response, the evaluation checks behavior-level properties such as intent, topic, dialog state, pedagogical strategy, and safe tutoring behavior. The framework is evaluated through an initial prototype case study in which one mathematics-help scenario is extended with seven metamorphic follow-up inputs and executed using automated pytest tests with trace information from the chatbot testing mode.*

Keywords: *metamorphic testing; behavior-driven development; educational chatbots; dialog systems; pedagogical consistency; robust artificial intelligence systems*

1. INTRODUCTION

Digital learning environments increasingly include educational chatbots that support explanation, practice, formative assessment, and access to learning materials. In this paper, the term educational chatbot denotes the conversational software system under test; the broader term educational dialog system is used only when discussing prior work that includes other forms of conversational agents. Unlike conventional information-retrieval tools, educational chatbots participate in a learning interaction and are expected to preserve both functional correctness and pedagogical appropriateness.

Testing such systems is challenging because learners rarely formulate their requests in a fixed or predictable way. The same intention may be expressed through paraphrases, informal language, spelling mistakes, incomplete messages, vague topic descriptions, or direct requests for complete solutions. A chatbot may therefore pass a predefined scenario and still fail when the same educational request is expressed differently. This creates an oracle problem: there is often no single correct response, but there are behavioral properties that should be preserved.

Behavior-driven development (BDD) scenarios are useful for specifying expected dialog flows and validating them against a dialog manager. However, traditional BDD tests often depend on predefined inputs and expected textual patterns. In educational dialog systems, this can lead to brittle tests that do not sufficiently capture semantic stability, intent preservation, or pedagogical consistency.

This paper proposes an extension of BDD testing with metamorphic relations. The central idea is to evaluate whether an educational dialog system preserves or appropriately adapts relevant behavioral properties under controlled transformations of learner input. Instead of checking only whether one input produces an expected response, the proposed approach checks behavior-level properties such as intent, topic,

dialog state, pedagogical strategy, and safe tutoring behavior.

The main contributions of this paper are:

- an extension of BDD-based dialog testing with metamorphic relations;
- a set of metamorphic relations suitable for educational chatbot scenarios;
- a behavior-oriented test oracle based on intent, topic, dialog state, and pedagogical consistency;
- a prototype evaluation showing how trace-based automated tests can reveal and support correction of robustness failures.

2. RELATED WORK

2.1. Behavior-Driven Testing and the Oracle Problem

Behavior-driven development (BDD) is commonly used to specify expected software behavior through scenarios written in semi-structured natural language. Such scenarios can improve communication among stakeholders and can also be executed as acceptance tests. Binamungu and Maro provide a systematic mapping study of BDD research and identify open issues related to evaluation methods, metrics, and maintenance of BDD artifacts [1]. In dialog systems, BDD scenarios are useful because they can describe user turns, expected responses, and dialog state transitions. However, educational dialog systems introduce additional complexity because the same learner intention may appear in many surface forms and a correct response does not necessarily have one fixed textual realization.

Metamorphic testing addresses situations in which it is difficult to define the exact expected output for a given input. Instead of checking only a single input-output pair, it defines relations between source test cases and follow-up test cases. Chen et al. describe metamorphic testing as a general approach for alleviating the oracle problem in software testing [2]. This idea is particularly relevant for educational dialog systems, where different response wordings may still preserve the same intent, topic, state, and pedagogical role.

2.2. Metamorphic Testing of Chatbots and LLM-Based Dialog Systems

Previous work has applied metamorphic testing to chatbots. Božić proposed an ontology-based approach that uses structured domain knowledge for test generation and output processing [3]. This work directly addresses the lack of a precise test oracle in chatbot testing. However, it is not focused on education-specific behavioral constraints such as learner uncertainty, scaffolding, hint-first guidance, or adaptation after repeated misunderstanding.

Recent research has also considered testing large language model (LLM)-based systems. Hou et al. provide a systematic literature review of LLMs for software engineering and show that software testing is one of the areas in which LLMs are actively explored [4]. Li et al. further discuss the evaluation of LLMs for software testing tasks and emphasize the need for systematic evaluation protocols [5]. In dialog systems, multi-turn interaction adds another level of difficulty because system behavior depends on previous turns and conversational state.

A closely related approach is MORTAR, a metamorphic multi-turn testing framework for LLM-based dialog systems [6]. MORTAR addresses the oracle problem by generating follow-up dialog test cases with dialog-level perturbations and metamorphic relations. Automated end-to-end testing of conversational agents has also been explored through user simulation and executable assertions, including robustness-related checks [9]. These works confirm the relevance of executable and metamorphic testing for modern dialog systems.

Educational chatbots introduce additional evaluation requirements. A recent systematic review shows that educational chatbot research covers different objectives, technologies, theoretical foundations, evaluation criteria, and impacts [7]. In addition, tutoring-oriented benchmarks emphasize that AI tutors should be evaluated not only by task-solving performance, but also by open-ended pedagogical capabilities [8]. Therefore, in educational settings, a chatbot should not only preserve semantic consistency, but also maintain or adapt pedagogical behavior, such as scaffolding, hint-first guidance, response to learner uncertainty, and avoidance of premature full-solution disclosure.

2.3. Research Gap

The reviewed literature shows that BDD provides a practical structure for specifying expected behavior, while metamorphic testing addresses the oracle problem in systems with variable outputs. Existing chatbot approaches primarily evaluate semantic or conversational consistency, and tutoring benchmarks evaluate pedagogical capabilities at the model level. The distinctive contribution of this paper is to connect executable BDD scenarios, controlled metamorphic input transformations, deterministic trace-based oracles, and explicit pedagogical constraints within one implementation-oriented testing workflow for educational chatbots.

3. PROBLEM STATEMENT

Existing BDD-based tests can verify whether an educational chatbot follows predefined dialog flows, such as asking for a topic, offering lesson

suggestions, or starting a quiz. However, such tests may not reveal whether the system remains stable when the same learner intention is expressed differently.

For example, the inputs "Explain fractions to me", "I do not understand fractions", and "Can you help me with fraction exercises?" may correspond to the same educational request. A robust educational dialog system should preserve the expected intent, topic, and pedagogical role across such variations. At the same time, some transformations should trigger controlled adaptation. For instance, when a learner explicitly expresses uncertainty, the chatbot should provide a more supportive or scaffolded response.

The main research question addressed in this paper is:

> How can BDD-based testing of educational chatbots be extended with metamorphic relations in order to evaluate semantic robustness and pedagogical consistency under controlled variations of learner input?

This question leads to three subquestions:

- Which input transformations are relevant for educational chatbot testing?
- Which behavioral properties should be preserved or adapted after such transformations?
- How can these relations be integrated into an executable testing framework?

4. PROPOSED FRAMEWORK

The proposed framework extends scenario-based BDD testing with metamorphic test generation and behavior-oriented evaluation. A base scenario defines an expected educational dialog flow, while metamorphic relations generate controlled variations of learner input. The system is then evaluated by checking whether relevant behavioral properties are preserved or appropriately adapted.

4.1. Framework Components

The framework consists of five main components:

- Base scenario repository — a collection of BDD-style dialog scenarios describing expected educational flows.
- Metamorphic input generator — a component that applies controlled transformations to learner messages.
- Dialog system under test — an educational chatbot or dialog manager.
- Behavior-oriented oracle — an evaluation mechanism that checks intent, topic, dialog state, and pedagogical constraints.
- Failure repository — a collection of violated relations and confirmed counterexamples used for regression testing.

4.2. Behavior-Oriented Test Oracle

The proposed oracle does not require a single fixed response. Instead, it checks whether relevant behavior-level properties are preserved after an input transformation. These properties include intent preservation, topic preservation, dialog-state consistency, pedagogical strategy, safe tutoring behavior, and appropriate adaptation to learner uncertainty.

A simplified representation of the oracle can be written as:

$$O_i = I_i \ni T_i \ni \Sigma_i \ni \Pi_i. \quad (1)$$

In this expression, I_i denotes intent preservation, T_i denotes topic preservation, S_i denotes state consistency, and P_i denotes pedagogical consistency. Equation (1) is a logical conjunction used to summarize pass conditions for a test case; it is not proposed as a mathematical model of learner behavior or chatbot dynamics. A relation passes only when all properties required for that specific relation are satisfied.

For implementation purposes, these properties can be operationalized through trace fields returned by the dialog system in testing mode. For example, intent preservation can be checked through a field such as intent, topic preservation through topic, state consistency through dialog_state, and pedagogical behavior through pedagogical_strategy. Direct requests for solutions can also be checked against safe tutoring constraints, such as hint-first or step-by-step guidance.

4.3. Test Execution Procedure

The procedure starts from a base BDD scenario and produces follow-up test cases by applying metamorphic relations. Responses are not compared as exact strings. For preservation relations, the source and follow-up traces must retain the specified intent, topic, and state fields. For adaptation relations, the follow-up trace must change in the prescribed direction, for example from an explanation strategy to remediation after repeated misunderstanding. Relation-specific assertions therefore define which fields must remain invariant and which fields may or must change.

Algorithm 1: Metamorphic testing of educational chatbot scenarios

1. Load a base BDD scenario with the source learner input and expected behavior.

2. Execute the chatbot in testing mode and record the response and trace fields.

3. Generate follow-up inputs using applicable metamorphic relations.

4. Execute the chatbot with each follow-up input using the relation-specific context derived from the same base scenario.

5. Check intent, topic, dialog state, and pedagogical constraints against the relation-specific oracle.

6. If a constraint is violated, log the source input, follow-up input, trace fields, response, and violated relation.

7. Add confirmed failures to the scenario repository as regression tests.

This procedure supports failure-driven improvement. Each violation becomes both diagnostic evidence and a new regression test for future versions of the dialog policy, prompt configuration, topic classifier, or pedagogical strategy module.

5. METAMORPHIC RELATIONS FOR EDUCATIONAL CHATBOTS

This section defines eight metamorphic relations for educational chatbot testing. Each relation consists of an input transformation and a relation-specific behavioral expectation. The set includes both preservation relations, such as paraphrasing and formality variation, and adaptation relations, such as learner uncertainty, direct solution requests, and repeated misunderstanding.

Table 1. *Proposed metamorphic relations for educational dialog testing*

Relation	Input transformation	Expected behavior
MR1	Paraphrase variation	Preserve learner intent and topic.
MR2	Word-order variation	Preserve the dialog-state transition.
MR3	Spelling/noisy input	Recognize intended topic if understandable.
MR4	Formality variation	Preserve the same pedagogical role.
MR5	Learner uncertainty	Provide more supportive/scaffolded response.
MR6	Equivalent topic request	Recommend same/equivalent lesson category.
MR7	Request for direct solution	Use safe tutoring and avoid premature full-solution disclosure.
MR8	Repeated misunderstanding	Change explanation strategy instead of repeating.

The initial prototype evaluation presented in Section 7 instantiates six of the eight relation categories. MR1 is represented by two follow-up variants (MR1a and MR1b), producing seven follow-up cases in total. MR2 and MR6 remain part of the general framework but are not exercised in the selected fractions-help scenario.

5.1. Example of a Metamorphic Relation

A base input may be:

"Explain linear equations to me."

A transformed input may be:

"I do not understand linear equations, can you help me?"

The expected response does not need to be textually identical. However, the chatbot should preserve the same topic and intent. In addition, because the learner expresses uncertainty, the response should include a more supportive explanation, a simple example, or a guiding question.

6. PROTOTYPE IMPLEMENTATION

The framework was implemented at the prototype level in a Python/Django educational chatbot for mathematics learning support. The chatbot exposes a REST API endpoint, and the automated tests use pytest together with Django REST Framework's APIClient. Each request contains a learner message, a test user identifier, and an explicit test_mode flag. Normal requests return only the learner-facing reply; testing-mode requests additionally return deterministic diagnostic trace data.

A simplified test request contains the following fields:

- message — the learner input;
- user_id — a test user identifier used to isolate dialog state between test cases;
- test_mode — a flag that enables trace output for testing purposes.

When test_mode is enabled, the trace is assembled after four processing steps: intent recognition, topic detection, dialog-state update, and pedagogical-policy selection. The returned fields include intent, topic, dialog_state, and pedagogical_strategy. For example, a fractions-help request should be classified as help_request, mapped to the topic fractions, placed in the explaining state, and assigned a guided explanation strategy. These fields provide an executable oracle without requiring one fixed response sentence.

The pytest suite represents the base BDD scenario as the source test case and parameterizes the follow-up inputs associated with the applicable metamorphic relations. Every relation uses an isolated user_id to prevent state leakage between tests. Preservation cases are evaluated from equivalent fresh contexts, whereas the history-dependent MR8 case first establishes the fractions topic with the base turn and then sends the repeated-misunderstanding input. The tests call the same endpoint in testing mode, read the returned trace, and apply relation-specific assertions. Equality assertions are used for properties that must be preserved, membership or

transition assertions are used for acceptable state changes, and negative assertions prevent unsafe behavior such as premature full-solution disclosure.

Testing mode is deterministic and does not call external large language model services. This design avoids network dependency, model nondeterminism, and external API quotas during regression testing. It also limits the scope of the present evaluation: the study validates the traceable dialog policy and classifier behavior, not the open-ended linguistic quality of a generative model.

7. PROTOTYPE CASE STUDY AND INITIAL EVALUATION

This section presents an initial prototype evaluation of the proposed framework on a mathematics-help scenario. The selected base scenario represents a learner request for help with fractions. The expected behavior is not defined as a fixed textual answer, but as a set of behavior-level properties: recognition of the help-seeking intent, preservation of the educational topic, maintenance of an appropriate dialog state, and selection of a suitable tutoring strategy.

Base scenario. Given a learner asks for help with fractions, when the chatbot processes the message in testing mode, then the expected intent is help_request, the expected topic is fractions, and the response should provide guided educational support rather than an unrelated answer or a complete unsupported solution.

The evaluation included one base scenario and seven metamorphic follow-up cases. The cases instantiate MR1 twice (MR1a and MR1b) and MR3, MR4, MR5, MR7, and MR8 once each. MR2 was not included because the selected inputs did not provide a meaningful word-order transformation beyond paraphrase, while MR6 requires a lesson-recommendation scenario rather than a help-with-fractions scenario. All eight executed tests, including the base case, were run automatically through the REST API. The first run passed seven tests; after correcting the defect revealed by MR3, the complete regression run passed eight of eight tests.

Table 2 presents the test inputs in English for linguistic consistency. The executable prototype used semantically equivalent Serbian inputs; the MR3 source contained an intentional spelling error and was not a proofreading mistake.

Table 2. *Initial metamorphic evaluation results for a fractions-help scenario*

MR	Learner input	Expected trace	Expected pedagogical behavior	Result
BASE	"Help me with fractions."	help_request, fractions, explaining	Provide guided explanation of the topic.	PASS
MR1a	"I do not understand fractions."	help_request, fractions, explaining	Preserve topic and provide scaffolded explanation.	PASS
MR1b	"Could you help me with fractions?"	help_request, fractions, explaining	Preserve help-seeking intent and provide guidance.	PASS
MR3	"Helpp me with fractionss."	help_request, fractions, explaining	Tolerate intentionally noisy input and preserve topic.	PASS
MR4	"Could you please explain fractions to me?"	help_request, fractions, explaining	Preserve pedagogical role despite polite wording.	PASS
MR5	"I am completely confused about fractions."	help_request, fractions, explaining	Provide additional scaffolding for learner uncertainty.	PASS
MR7	"Just give me the answer to the fractions problem."	solution_request, fractions, guided_problem_solving	Use a hint-first strategy instead of full-solution disclosure.	PASS
MR8	"I still do not understand."	clarification_request, fractions, remediation	Use an alternative explanation strategy.	PASS

The first test run passed seven of eight executed cases. MR3 failed because the deterministic classifier did not tolerate a minor spelling error in the Serbian help-seeking input, even though the intended topic remained understandable. The trace showed that the expected help_request classification was not produced. After a typo-tolerant recognition rule was added, the complete suite passed eight of eight cases. The confirmed counterexample was retained as a regression test, demonstrating a direct path from relation violation to diagnosis, correction, and future protection.

The same workflow can be applied to additional educational scenarios, including linear-equation explanations, homework assistance, incorrect quiz answers, lesson recommendations, and repeated requests after failed understanding. Such extensions would increase relation coverage: a lesson-recommendation scenario would exercise MR6, and deliberately reordered but semantically equivalent requests would exercise MR2. Future evaluations should also use multiple topics and learner profiles and report per-relation pass rates and defect-discovery counts.

7.1. Failure Classification and Development Feedback

The purpose of the framework is not only to count passed and failed tests, but also to support diagnosis and iterative improvement. Violations of

metamorphic relations can be classified as paraphrase robustness failures, noisy-input failures, uncertainty-adaptation failures, premature-solution failures, or remediation failures. Each category suggests a corresponding development action, such as adding paraphrase examples, typo-tolerant recognition, uncertainty detection, hint-first policy checks, or alternative explanation paths.

The observed MR3 failure corresponds to the category of insufficient tolerance to noisy learner input. The corrective action was the addition of typo-tolerant recognition for the Serbian help-seeking marker in the deterministic test-mode classifier. The practical value of the method is that the detected problem is not only recorded as a failed test, but also converted into a regression case that protects the same behavior in future versions of the chatbot. This example shows that metamorphic testing is useful not only for verification, but also for failure-driven improvement of educational dialog behavior.

8. DISCUSSION

The proposed approach addresses a limitation of traditional scenario-based chatbot testing. BDD scenarios validate expected flows but do not systematically test the variability of learner language. General metamorphic chatbot testing addresses semantic consistency, whereas the present framework additionally treats pedagogical behavior as part of the oracle. A technically consistent response can therefore still fail when it reveals a full solution too early, ignores uncertainty, or repeats an ineffective explanation.

This pedagogical layer is the main difference from approaches that check only intent or response equivalence. Preservation relations verify semantic stability, while adaptation relations require a controlled change in tutoring strategy. The trace-based implementation makes these expectations executable and separates the tested behavior from the exact wording of the reply.

The MR3 result illustrates the practical value of the method. The test did not merely report a mismatch; it localized the failure to noisy-input recognition, motivated a specific classifier correction, and created a permanent regression case. A larger repository of such counterexamples could guide changes in dialog policy, prompt configuration, topic recognition, and remediation strategies.

Although this paper focuses on educational chatbots, the same principle may be relevant to other adaptive artificial intelligence systems whose behavior is evaluated through invariants, safety constraints, or consistency relations rather than exact outputs. Possible future applications include intelligent tutoring in other domains, educational assessment tools, and human-machine interaction scenarios in vehicle-related training and support.

9. LIMITATIONS AND FUTURE WORK

This paper presents a conceptual and implementation-oriented framework supported by an initial prototype evaluation. The case study demonstrates how metamorphic relations can be executed through a chatbot testing mode, but the evaluation is limited to one mathematics-help scenario and a small set of follow-up inputs. Future work should include a larger scenario repository, additional educational topics, automatically generated metamorphic variants, and quantitative metrics such as pass rate, violation rate per relation, flakiness, and defect discovery rate.

Another limitation is that pedagogical behavior was evaluated through explicit trace fields and rule-based checks. Future work should refine the classification of pedagogical failures, including lack of scaffolding, premature solution disclosure, inadequate recommendation, failure to recognize learner uncertainty, and repetition without adaptation.

Large language models may also be used to generate paraphrases, noisy variants, and candidate failure explanations. However, such use should be combined with explicit rules, regression tests, and human validation, rather than relying only on LLM-as-a-judge evaluation.

10. CONCLUSION

This paper proposed an extension of BDD-based testing with metamorphic relations for robust testing of educational chatbots. The approach moves from fixed input-output expectations to behavior-oriented evaluation under controlled variations of learner input. Instead of requiring identical responses, the framework checks whether intent, topic, dialog state, pedagogical strategy, and safe tutoring behavior are preserved or adapted as required by each relation.

The prototype evaluation showed how trace information from a chatbot testing mode can support automated metamorphic tests. One base mathematics-help scenario was extended with seven follow-up inputs, and the resulting test suite was executed using pytest. The initial MR3 failure also illustrated how a violated metamorphic relation can lead to a concrete improvement and become a regression test.

By combining BDD scenarios, metamorphic relations, and pedagogical behavior constraints, the proposed framework provides a practical basis for more robust and semantically meaningful testing of artificial intelligence-supported educational chatbots.

REFERENCES

[1] Binamungu, L. P. & Maro, S. (2023). Behaviour driven development: A systematic mapping study. *Journal of Systems and Software*, *203*, Article 111749. doi:10.1016/j.jss.2023.111749

[2] Chen, T. Y., Kuo, F.-C., Liu, H., Poon, P.-L., Towey, D., Tse, T. H. & Zhou, Z. Q. (2018). Metamorphic testing: A review of challenges and opportunities. *ACM Computing Surveys*, *51*(1), Article 4, 4:1–4:27. doi:10.1145/3143561

[3] Božić, J. (2022). Ontology-based metamorphic testing for chatbots. Software Quality Journal, *30*(1), 227–251. doi:10.1007/s11219-020-09544-9

[4] Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J. & Wang, H. (2024). Large language models for software engineering: A systematic literature review. *ACM Transactions on Software Engineering and Methodology*, *33*(8), Article 220. doi:10.1145/3695988

[5] Li, Y., Liu, P., Wang, H., Chu, J. & Wong, W. E. (2025). Evaluating large language models for software testing. *Computer Standards & Interfaces*, *93*, Article 103942. doi:10.1016/j.csi.2024.103942

[6] Guo, G., Aleti, A., Neelofar, N., Tantithamthavorn, C., Qi, Y. & Chen, T. Y. (2026). MORTAR: Multi-turn metamorphic testing for LLM-based dialogue systems. *IEEE Transactions on Software Engineering*, in press. doi:10.1109/TSE.2026.3701230

[7] Debets, T., Banihashem, S. K., Joosten-Ten Brinke, D., Vos, T. E. J., Maillette de Buy Wenniger, G. & Camp, G. (2025). Chatbots in education: A systematic review of objectives, underlying technology and theory, evaluation criteria, and impacts. *Computers & Education*, *234*, Article 105323. doi:10.1016/j.compedu.2025.105323

[8] Macina, J., Daheim, N., Hakimi, I., Kapur, M., Gurevych, I. & Sachan, M. (2025). MathTutorBench: A benchmark for measuring open-ended pedagogical capabilities of LLM tutors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 204–221. doi:10.18653/v1/2025.emnlp-main.11. Available at: arxiv.org/abs/2502.18940

[9] De Lara, J., Del Pozzo, A., Guerra, E. & Sánchez Cuadrado, J. (2026). Automated end-to-end testing for conversational agents. *Journal of Systems and Software*, *233*, Article 112685. doi:10.1016/j.jss.2025.112685