Vol 42, No 2 (2025): October 2025

Jurnal Penelitian Pendidikan

https://journal.unnes.ac.id/journals/JPP

An Innovative Approach to Automatically Generating Derivative Questions for Calculus with Rule-based Algorithmic Model

Trisnani Widowati[1*], Septian Eko Prasetyo[2], Anan Nugroho[2], Ade Novi Nurul Ihsani[1], Alfian Ardhiansyah[2], Ajeng Rahma Sudarni[2], Mulil Khaira[2], Clarita Aprilliani[1], Anik Maghfiroh[1]

[1]Department of Family Welfare Education, Faculty of Engineering,
Universitas Negeri Semarang, Indonesia
[2]Electrical Engineering Department, Faculty of Engineering,
Universitas Negeri Semarang, Indonesia

*Correspondent Author: niwid@mail.unnes.ac.id

Abstract
This study aims to develop an automated system for generating derivative function questions to enhance learning and assessment in calculus education. The system uses a rule-based algorithmic approach to generate questions covering various derivative rules, including constant, power, sum, difference, product, quotient, and chain rules, with randomly generated coefficients and exponents. The system's performance was evaluated by measuring the accuracy of the generated questions, achieving a validity rate of 95.63%, and execution time, with an average of 0.00115 milliseconds per question, indicating high efficiency and effectiveness. The results show that the system successfully generates valid questions across various derivative rules, offering significant contributions to both students and educators in the context of calculus education. The system can be further developed by adding additional derivative rules and integrated into larger educational platforms to support a more comprehensive learning experience.

Keywords: automated item generation, algorithmic question generation, calculus, derivative function, rule-based system

INTRODUCTION

The advancement of technology in the last decade has significantly transformed various sectors, including communication, industry, finance, and particularly education [1], [2]. In modern educational practices, the integration of technology is not only used to deliver content but also to support assessment and evaluation processes, thereby enhancing both the efficiency and the effectiveness of learning [3]. One such technological innovation is Computer-Assisted Assessment (CAA), which allows for fast and objective grading of student answers [4]. While CAA traditionally focuses on scoring responses, recent developments have led to the emergence of Automated Item Generation (AIG), a field of research that seeks to develop systems capable of automatically generating diverse and structured questions [5].

Several studies have explored AIG across various domains, showcasing its potential in revolutionizing assessment and enhancing question diversity. For example, Kenneth et al. conducted a study on AIG in the medical field, focusing on its application in creating valid and diverse assessment items for medical education [6]. In the field of language, another study utilized Natural Language Processing (NLP) to generate questions by extracting keywords from the material, demonstrating AIG's versatility in linguistic contexts [7]. Additionally, Gierl et al. developed a specific method for AIG that has been widely adopted in mathematics, leveraging a semi-automated framework model. This approach, however, requires specialized skills in stem development to generate the desired questions effectively, highlighting the need for expert input in the process [8]. Another noteworthy contribution to AIG is the development of the Psychometric Item Generator (PIG), introduced as a solution for psychological scale development. PIG is a natural language processing algorithm based on the GPT-2 model, capable of generating large-scale, human-like items with minimal effort and no prior coding skills [9].

Research has highlighted the limitations of Automatic Item Generation (AIG) systems in complex mathematical problem. Several studies have attempted to develop mathematical approaches; however, based on our review, these efforts are still limited to simple calculations. For example, a study conducted by Heru Widiatmo et al [10]. developed

a mathematics AIG system using a predefined template or stem-based model. While this approach can generate items, it presents challenges for users who lack the expertise to create high-quality questions, accurate answers, and effective distractors. The model has also been developed by Gierl and colleagues, who attempted to use complex mathematical formulas through a stem-based question method [11].

On the other hand, Prasetyanto et al [12] attempted to develop a mathematical AIG model for geometry and algebra. The system developed is capable of handling complex mathematical computations. However, the main issue remains that the model still relies on a template-based stem system, where users must understand how to utilize question stems, create answer distractors, and define correct answers. This model is suitable for generating parallel mathematics questions, where the difficulty level can be well controlled. However, the variety of generated questions is limited due to the constraints of user-defined templates.

These studies collectively demonstrate significant progress in automatic question generation, particularly in the field of mathematics. Each developed approach has been able to address existing problems effectively. However, the models and studies that have been developed so far are still unable to handle complex mathematical problems such as derivatives, which involve high-level mathematical complexity. This presents a particular challenge in developing automated question generation systems for calculus problems. Furthermore, a new challenge in automatic math problem generation lies in developing a fully automated framework model, where users do not need to design question stems in advance. Ideally, the system should be able to compute and generate problems automatically based on mathematical rules.

The main objective of this study is to design and evaluate an automated system capable of generating a wide variety of derivative mathematics problems. In addition, the system will produce credible and contextually relevant multiple-choice distractors. The developed model adopts an automated framework approach, eliminating the need to use stem templates for question generation. This innovative approach aims to provide valuable support to educators and students by facilitating efficient, scalable, and personalized assessments, thereby enhancing the overall experience of calculus learning.

METHOD

This study develops an automated system aimed at generating derivative questions quickly and diversely, while also producing credible and relevant multiple-choice distractors. The system is developed using a rule-based algorithmic approach, which allows customization and development of questions according to the needs of calculus material. In Automated Item Generation (AIG), there are two main models: automatic and semi-automatic [13]. The automatic model, like the one developed in this study, heavily relies on rule-based algorithms to generate questions. This model automates the entire process by utilizing predefined mathematical rules, such as the power rule, product rule, and chain rule, to compute the derivative and structure the questions. The automatic system is designed to generate mathematically valid questions and plausible answer choices efficiently, without human intervention. The algorithm employed in this study is presented in Figure 1.

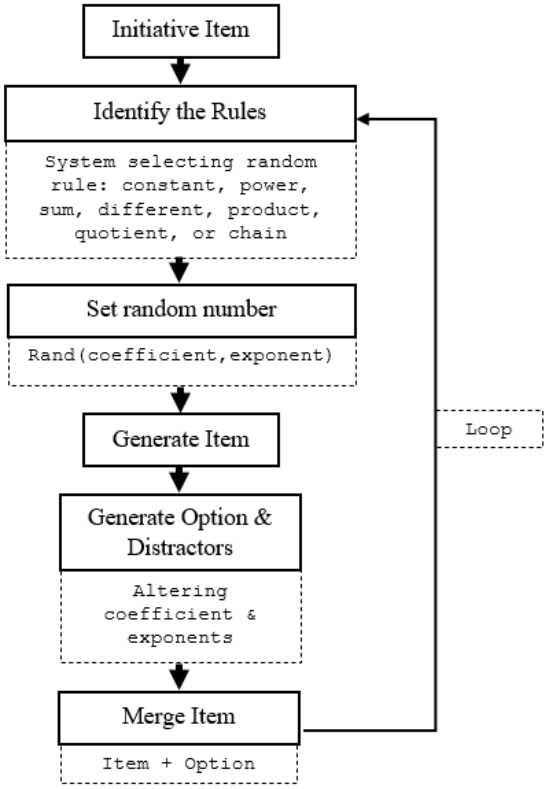

Figure 1. Algorithm Process

Algorithm Process

The algorithm for generating automatic derivative questions as presented in Figure 1 can be explained as follows:

1. ***Initiative Item***: Initially, the system initializes the number of questions as specified by the user. Subsequently, the system performs a loop based on the defined number of questions.
2. ***Identify the Rules***: The system randomly selects a derivative function to be executed. The selection is based on derivative rules, including the constant rule, power rule, sum rule, difference rule, product rule, quotient rule, and chain rule.
3. ***Set random number***: Once the function has been selected, the system automatically generates a function to produce random numbers. These numbers are assigned as variables: the coefficient, which is the numeric factor preceding a variable in a term, and the exponent, which indicates the power to which the variable is raised. The range of coefficient values is from 1 to 10, while the exponent values range from 1 to 5. The combination of these values is processed to construct the question, answer options, and distractors in the generated item.
4. ***Generate item***: During this stage, the algorithm transforms the previously defined variables into a question. The resulting item is constructed based on the specific function or derivative rule that was randomly selected previously by the system.
5. ***Generate option and distractors***: In multiple-choice question development, there are several commonly used terminologies, including options and distractors [14]. The options consist of both the correct answer and incorrect answers. The correct answer is obtained from the functional calculation process of the previously selected function, which returns an accurate result. On the other hand, distractors are the incorrect answer choices. The creation of distractors follows a set of complex rules designed to effectively mislead or challenge the test taker. One of the distractor generation methods used in this study includes: multiplying the exponent with the coefficient, multiplying the coefficient with the exponent, retaining a constant term (i.e., not differentiating it), not reducing the exponent by one, failing to multiply the exponent with the coefficient, and other techniques.
6. ***Merge item***: After the question and options have been generated, the next step is to combine both elements into a complete multiple-choice question. A marker will be provided for the correct answer to assist the question developer in the evaluation process.
7. ***Loop***: All processes in the algorithm will be repeated to the number of question that ware initialized earlier.

Derivative Rules

The derivative of a function represents the rate of change of the function with respect to its variable. The

mathematical definition of a derivative can be expressed in formula 1. The derivative can also be interpreted as the slope of the tangent line to the graph of the function at a specific point, or as the rate of change of one quantity with respect to another variable [15].

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \quad (1)$$

Where $f'(x)$ is the derivative of the function $f(x)$ with respect to the variable x. The limit $lim\ h \to 0$ represents the change in the function as h (the change in x) becomes infinitesimally small and approaches zero. The value h represents a small change in the variable x, often referred to as a small increment in x or the difference in input values.

Table 1. Derivative Rules

Rule	Formula
Constant	$f(x) = c$ $f'(x) = 0$
Power	$f(x) = ax^n$ $f'(x) = n \,.\, ax^{(n-1)}$
Sum	$f(x) = g(x) + h(x)$ $f'(x) = g'(x) + h'(x)$
Difference	$f(x) = g(x) - h(x)$ $f'(x) = g'(x) - h'(x)$
Product	$f(x) = g(x) * h(x)$ $f'(x) = g'(x).h(x) + g(x).h'(x)$
Quotient	$f(x) = \frac{g(x)}{h(x)}$ $f'(x) = \frac{g'(x).h(x) - g(x).h'(x)}{h(x)^2}$
Chain	$f(x) = g(h(x))$ $f'(x) = g'(h(x)) \,.\, h'(x)$

However, in practice, simpler derivative formulas are used to facilitate calculations. These shortcut formulas are the result of simplifying the limit definition and allow us to compute derivatives of various functions more efficiently without the need to use the limit process [16]. In developing the automated system for generating derivative questions, various fundamental rules of derivative are employed to ensure the mathematical validity and diversity of the questions. These rules form the foundation of the algorithm and guide the process of computing the derivatives. The rules used in this study as shown in Table 1. Where *f (x)* is the original or base function, which is the function to be differentiated. Meanwhile, *f'(x)* is the first derivative of *f.(x)*, which is the result of the differentiation process with respect to the variable *x*.

Distractor Generator

In the context of automatically generating derivative questions, the presence of distractors (incorrect answer choices that seem plausible) is an essential element for assessing students' conceptual understanding [17]. This process is designed using a rule-based approach, which is structured based on common patterns of errors students make when solving derivative questions. The model first identifies the type of derivative function used, then generates the correct answer using standard derivative rules. Once the correct answer is obtained, the system generates distractors by applying common errors that have been previously classified, such as:

1. **Chain rule errors**: Distractors are created by neglecting the derivative of the inner part of a composite function.
2. **Neglecting product or quotient rules**: The system constructs options by treating the function as regular multiplication, without applying the product rule.
3. **Incorrect derivative of constants or coefficients**: This includes errors in multiplying constants or incorrectly differentiating zero constants.
4. **Basic algebraic mistakes**: Such as errors in simplifying expressions before or after the derivative process.
5. **Incorrectly identifying function forms**: For example, treating the exponential function e^{x2} as a regular

polynomial function and differentiating it like x^2 , yielding $2x$ without considering the chain rule.

6. **Using inappropriate rules**: For instance, applying the product rule to a function that should be treated as a single function, or vice versa, leading to overly complex or irrelevant derivative forms.
7. **Direct division of derivatives**: In the case of fractions, students often incorrectly divide the derivatives of the numerator and denominator. The problem should be solved using the quotient rule.

By expanding the basis of error rules as previously explained, the generated distractors will be more varied and reflect the real challenges students face in understanding the concept of derivatives. This also enhances the validity of the questions as a measurement tool for learning.

Evaluation Method

The model evaluation is conducted to ensure that the developed system can generate derivative questions along with their answer choices both mathematically correct and efficiently. Since this system uses a rule-based algorithmic approach, the evaluation process is focused on two main aspects: the mathematical correctness of the model's output and the system's performance in generating questions automatically. The correctness of the derivative results and distractors needs to be verified symbolically to align with the principles of calculus, while the system's performance is important to assess in terms of time efficiency and scalability in a technology-based learning environment. Therefore, the evaluation is carried out in two phases: validation of the question results using symbolic computation libraries, and performance testing of the system in generating questions continuously.

Validation of Question Output

After the derivative questions are generated by the model, the first phase of evaluation is mathematical validation. This process is conducted to ensure that the answers produced by the system are correct according to calculus and align with the applicable derivative rules. The primary tool used in this validation is SymPy, a Python library for symbolic computation that enables automatic verification of derivative results [18]. By using SymPy, each question generated by the model is recalculated to verify the correctness of the derivative results and ensure that no errors occur in the application of the derivative rules. The tested functions include polynomials, exponential, trigonometric, and other composite functions. If the results calculated by the model match those computed by SymPy, the question is considered valid.

Manual validation by humans or experts was not feasible due to the large number of questions generated by the system. Evaluating each question manually would be highly time-consuming and prone to human error [19]. Therefore, automated validation using SymPy was chosen as a scalable, reliable, and efficient solution to handle the volume and complexity of the generated derivative questions. After validating the question results using SymPy, the next step is to measure the percentage of valid questions from all the tested questions. The validity of the questions is measured based on whether the derivative results generated by the model align with the results computed by SymPy. The calculation of the validity percentage is performed using formula 2

$$V = \frac{(Number\ of\ Valid\ Question)}{Total\ Number\ of\ Question}\ x\ 100 \qquad (2)$$

Where Number of Valid Questions is the count of questions that passed the validation check using SymPy, and Total Number of Questions is the total number of questions tested in the validation process.

Performance Testing

In addition to the validation of the correctness of the questions, the evaluation also focuses on the model's performance in automatically generating derivative questions. The performance testing aims to assess the system's efficiency when used to generate questions on a large scale. Execution time measures the average time required by the system to generate a complete set of questions, from initiating the number of questions to be developed, to producing multiple-choice questions complete with answers and distractors. This testing is conducted on a set of functions with varying levels of complexity, such as the constant rule, power rule, sum rule, difference rule, product rule, quotient rule, and chain rule.

To obtain objective results that represent real-world usage, the testing is performed within a standardized system environment on online server. Each type of function is tested with varying numbers of questions (for example, 100, 500, and 1000 questions generated by the system) to evaluate the consistency of the system's performance under different workload scenarios. The time taken is calculated from the moment the user provides parameters for the number of questions until all components of the questions, including the stem, answer key, and distractor options, are successfully generated.

To precisely measure the execution time, this research utilizes a benchmarking method based on timestamps by leveraging the performance.now() function from the JavaScript Web API. This function provides high-resolution time

measurement (in milliseconds to microseconds), allowing accurate recording of the start and end times of the question generation process [20]. The difference between these two timestamps is used to determine the execution duration of the algorithm. The execution duration is calculated by finding the difference between these two timestamps using the following formula 3.

$$D = T_{end} - T_{start} \quad (3)$$

Where D is the execution duration (in milliseconds).T_{Start} is the timestamp recorded at the beginning of the process. T_{End} is the timestamp recorded at the end of the process. This method was chosen for its ability to provide stable and detailed measurement results, particularly in fast computational processes like rule-based question generation.

RESULTS AND DISCUSSION

In this section, the outcomes of the experimental evaluation of the proposed algorithmic model for automatic derivative question generation are presented. The evaluation focuses on two primary aspects: the accuracy of the generated questions and the performance of the system in terms of efficiency and computational time. The accuracy validation examines how well the generated questions align with the expected mathematical principles and standards, while the performance evaluation measures the system's ability to generate a large number of questions quickly, under varying number of question and levels of complexity.

The results of both evaluations provide valuable insights into the strengths and limitations of the model, as well as its practical applicability for educational purposes. Additionally, the discussion delves into the implications of these results, addressing the model's potential for integration into automated learning platforms and its contribution to enhancing calculus questions generation in a scalable and efficient manner.

Accuracy Validation Result

The validation of the automatically generated derivative questions was conducted using three testing scenarios: the generation of 100 questions, 500 questions, and 1000 questions. In each scenario, the generated questions were classified based on the specific differentiation rules applied, including the constant rule, power rule, sum rule, difference rule, product rule, quotient rule, and chain rule. For each scenario, the system computationally analyzed how many questions were successfully generated for each rule, how many were valid, and how many were invalid. The validity assessment was performed using the automated evaluation method described earlier, which utilizes SymPy to verify the mathematical correctness of each derivative result.

The evaluation results for the generation of 100 questions are presented in Table 2. The outcomes for the 500-question generation are shown in Table 3, while the evaluation of the 1000-question generation is summarized in Table 4. This multi-scenario approach provides a comprehensive understanding of the system's performance across different volumes of question generation, highlighting its consistency, scalability, and robustness in producing mathematically valid and pedagogically sound derivative questions.

The results of the three testing scenarios are presented in Tables 2, 3, and 4, respectively. In the first scenario, as shown in Table 2, the system successfully generated 100 derivative questions distributed across various differentiation rules. After conducting mathematical validation, it was found that 4 questions were invalid based on differentiation rule violations. Consequently, the system achieved a validity rate of 96%, with 96 questions confirmed as mathematically correct. A similar pattern is observed in the second scenario presented in Table 3, where the system generated 500 derivative questions. Out of these, 478 questions were validated as correct, while 22 questions were identified as invalid. This results in a slightly lower but still high validity rate of 95.6%. In the final scenario, shown in Table 4, the system generated a larger batch of 1000 questions. The mathematical validation process identified 953 valid questions and 47 invalid questions, resulting in a validity percentage of 95.3%. This consistency across increasing question volumes demonstrates the robustness and reliability of the system in generating mathematically accurate derivative questions at scale.

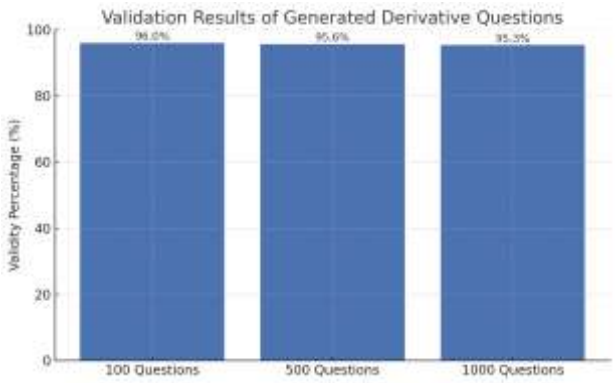

Figure 2. Validation Result for 3 Scenario

These results indicate that, on average, 95.63% of the derivative questions generated by the system were mathematically valid across all three testing scenarios. This consistently high validity rate demonstrates the reliability and effectiveness of the proposed rule-based algorithm in producing accurate and well-structured calculus questions at various scales (see Figure 2).

Performance Evaluation Result

System performance was evaluated by measuring the average execution time required to generate complete question sets under varying workloads, namely 100, 500, and 1000 questions. These workloads were designed to cover all seven derivative rule categories to ensure comprehensive evaluation of the generation algorithm. The performance tests were conducted in a controlled environment using a cloud server with minimal specifications, ensuring that the evaluation reflected the baseline performance without relying on high-end computational resources. In the scenario involving 100 questions, the system achieved a total execution time of 0.140 milliseconds, with an average time per question of 0.00140 milliseconds (Table 2). For 500 questions, the system maintained efficient performance with a total time of 0.572 milliseconds and an average of 0.00114 milliseconds per question (Table 3). In the 1000-question scenario, the total time was 1.150 milliseconds, averaging 0.00115 milliseconds per question (Table 4).

The consistency of the average execution time across increasing workloads suggests that the algorithm scales effectively, maintaining low latency even as the number of generated questions grows. This indicates a high level of computational efficiency, likely due to the optimization in the algorithm's rule-matching and question formatting procedures. Notably, the average execution time per question slightly decreases as the workload increases, before stabilizing, which may indicate the presence of internal optimizations such as reusable computation structures or memory caching mechanisms. Compared to similar systems referenced in prior research, which report per-item generation times in the range of milliseconds to seconds, this system demonstrates significantly higher efficiency. These findings underscore the feasibility of deploying the algorithm in real-time educational applications, such as adaptive testing platforms or large-scale online examinations, where both speed and scalability are critical.

Discussion

The evaluation of the proposed rule-based algorithm highlights its effectiveness in generating mathematically valid derivative questions at various scales. Across all three testing scenarios—100, 500, and 1000 questions—the system consistently maintained a high validity rate, averaging 95.63% (see Tables 2–4). This indicates that the algorithm reliably applies the derivative rules, even as the workload increases. Minor variations in validity percentages suggest occasional rule misapplications, which remain within acceptable margins and do not significantly compromise the system's overall reliability. These results affirm the system's robustness in producing structurally and mathematically sound problems.

In terms of performance, the system demonstrated efficient execution with low latency in all scenarios. Even with 1000 questions, the total execution time remained slightly above one millisecond, with per-question averages remaining consistent (Tables 2–4). This stability reflects the efficiency of the algorithm's internal processing, including rule matching and symbolic derivative. The slight decrease in average execution time per question as workload increases suggests the presence of internal optimizations, such as computation reuse or lightweight memory handling.

Table 2. Evaluation Results of Validity and Algorithm Performance to Generate 100 Questions

Rule	Question Generated	Valid Questions	Invalid Questions	Execution Time (ms)	Average Execution Time (ms)
Constant Rule	15	14	1	0.020	0.00133
Power Rule	15	15	0	0.022	0.00147
Sum Rule	15	14	1	0.019	0.00127
Difference Rule	14	14	0	0.020	0.00143
Product Rule	14	13	1	0.025	0.00179
Quotient Rule	14	13	1	0.020	0.00143
Chain Rule	13	13	0	0.015	0.00115
Total	100	96 (96%)	4 (4%)	0.140	0.00140

Table 3. Evaluation Results of Validity and Algorithm Performance to Generate 500 Questions

Rule	Question Generated	Valid Questions	Invalid Questions	Execution Time (ms)	Average Execution Time (ms)
Constant Rule	73	68	5	0.050	0.00068
Power Rule	72	70	2	0.075	0.00104
Sum Rule	71	70	1	0.084	0.00118
Difference Rule	71	68	3	0.078	0.00110
Product Rule	71	67	4	0.103	0.00145
Quotient Rule	71	68	3	0.104	0.00146
Chain Rule	71	67	4	0.078	0.00110
Total	500	478 (95.6%)	22 (4.4%)	0.572	0.00114

Table 4. Evaluation Results of Validity and Algorithm Performance to Generate 1000 Questions

Rule	Question Generated	Valid Questions	Invalid Questions	Execution Time (ms)	Average Execution Time (ms)
Constant Rule	143	133	10	0.069	0.00048
Power Rule	143	143	0	0.134	0.00094
Sum Rule	143	140	3	0.166	0.00116
Difference Rule	143	137	6	0.172	0.00120
Product Rule	143	133	10	0.232	0.00162
Quotient Rule	143	135	8	0.229	0.00160
Chain Rule	142	132	10	0.148	0.00104
Total	500	953 (95.3%)	47 (4.7%)	1.150	0.00115

When compared to prior models, including those developed in works referenced by [8], [10]–[12], the proposed algorithm presents several notable advancements. First, its ability to both generate and evaluate derivative questions simultaneously allows for faster data generation pipelines, which is particularly useful for testing, training, or educational deployment. Unlike earlier models that required predefined question stems or relied on manual formatting, this system adopts a fully automated framework. It uses a library of predefined mathematical transformation rules to construct questions dynamically, thus eliminating the need for hand-crafted question templates.

Moreover, this flexible framework supports the generation of diverse questions types, ranging from basic derivative computation to more complex chain rule or product rule applications. The ability to introduce variability in question format not only supports better pedagogical diversity but also reduces the likelihood of student overfitting to repeated question patterns. This adaptability is a key differentiator that enhances the system's utility in modern educational contexts.

From a pedagogical standpoint, the model aligns well with current trends in adaptive learning and intelligent tutoring systems. Its combination of speed and correctness makes it suitable for integration into large-scale e-assessment platforms where automated question generation must meet strict standards of mathematical accuracy. Additionally, its capability to produce varied and structurally sound problems contributes to fairer and more effective assessment by minimizing predictability and encouraging deeper conceptual understanding among students.

In conclusion, the proposed rule-based algorithm demonstrates not only technical robustness and computational efficiency but also educational relevance. Its dual strengths—accurate symbolic computation and automated question generation—make it a promising solution for deployment in a variety of educational technologies. Future work may involve expanding the rule library to cover more advanced calculus topics, integrating semantic checking for problem instructions, and refining the feedback mechanism for incorrect answers to further enhance its role as an intelligent educational assistant.

Future Research

Although the developed system has demonstrated high accuracy and efficiency in automatically generating derivative questions, there is still room for further development. Future research can focus on expanding the scope of mathematical rules to include more complex calculus topics such as partial derivatives, limits, or integrals, allowing the system to be used more broadly in advanced mathematics learning contexts. Additionally, integrating semantic validation of the generated questions is an important direction to ensure that each question is not only structurally and mathematically valid, but also aligned with the intended learning context. Developing an automatic feedback mechanism for incorrect answers could further enhance the system into an intelligent tutor capable of providing direct guidance to students. From a technical perspective, further optimization of memory processing and execution time is also necessary, particularly to accommodate real-world implementation scenarios on a large scale, such as online assessment systems or adaptive learning platforms. Lastly, evaluating the effectiveness of this system in actual learning environments—such as through experimental studies with students and educators—could provide deeper insight into its pedagogical impact and readiness for deployment in educational settings.

CONCLUSIONS

The proposed system demonstrated robust accuracy and computational efficiency in generating derivative questions across a range of test scenarios. With an average validity rate of 95.63% in batches of 100, 500, and 1000 questions, the algorithm consistently produced mathematically correct problems with minimal error rates. This high level of accuracy confirms the system's capability to reliably apply derivative rules and symbolic transformations, even under increasing workloads.

Performance evaluations further affirm the system's strength, as it maintained impressively low execution times—ranging from 0.00114 to 0.00140 milliseconds per question—regardless of the scale. This indicates not only algorithmic efficiency but also excellent scalability, positioning the system as a viable solution for high-demand, real-time educational environments. The low latency and consistent processing times are critical for applications such as adaptive learning systems, online assessments, and intelligent tutoring platforms, where responsiveness and accuracy are essential.

Beyond its technical merits, the system contributes meaningful pedagogical value by enabling the automated generation of diverse and structurally valid derivative questions. This flexibility reduces the reliance on manually created problem sets and opens opportunities for personalized learning experiences through dynamically generated content.

In summary, the algorithm offers a powerful combination of correctness, speed, and scalability. These attributes make it a strong candidate for integration into educational technologies aimed at enhancing mathematics instruction, supporting large-scale testing, and promoting individualized learning. Future work may explore extending the system

to broader mathematical domains, incorporating semantic validation for richer question types, and enhancing its adaptability to curriculum standards across different educational contexts.

REFERENCES

[1] M. L. Owoc, A. Sawicka, and P. Weichbroth, "Artificial Intelligence Technologies in Education: Benefits, Challenges and Strategies of Implementation," 2021, pp. 37–58.

[2] M. Javaid, A. Haleem, R. P. Singh, and A. K. Sinha, "Digital economy to improve the culture of industry 4.0: A study on features, implementation and challenges," *Green Technologies and Sustainability*, vol. 2, no. 2, p. 100083, May 2024.

[3] J. Suárez-Álvarez, R. Fernández-Alonso, F. García-Crespo, and J. Muñiz, "The Use of New Technologies in Educational Assessments: Reading in a digital world," *Papeles del Psicólogo - Psychologist Papers*, vol. 43, no. 1, 2022.

[4] J. C. Paiva, J. P. Leal, and Á. Figueira, "Automated Assessment in Computer Science Education: A State-of-the-Art Review," *ACM Transactions on Computing Education*, vol. 22, no. 3, pp. 1–40, Sep. 2022.

[5] B. Das, M. Majumder, S. Phadikar, and A. A. Sekh, "Automatic question generation and answer assessment: a survey," *Res Pract Technol Enhanc Learn*, vol. 16, no. 1, p. 5, Dec. 2021.

[6] K. D. Royal, M.-W. Hedgpeth, T. Jeon, and C. M. Colford, "Automated Item Generation: The Future of Medical Education Assessment," *EMJ Innovations*, pp. 88–93, Jan. 2018.

[7] C. A. Nwafor and I. E. Onyenwe, "An Automated Multiple-Choice Question Generation using Natural Language Processing Techniques," *International Journal on Natural Language Computing*, vol. 10, no. 02, pp. 1–10, Apr. 2021.

[8] M. J. Gierl, H. Lai, and V. Tanygin, *Advanced Methods in Automatic Item Generation*. Routledge, 2021.

[9] F. M. Götz, R. Maertens, S. Loomba, and S. van der Linden, "Let the algorithm speak: How to use neural networks for automatic item generation in psychological scale development.," *Psychol Methods*, vol. 29, no. 3, pp. 494–518, Jun. 2024.

[10] H. Widiatmo and A. Suryanto, "Developing Automatic Item Generation," in *2022 International Conference on Innovation in e-ISSN 2963-2870 Open and Distance Learning*, 2022, vol. 3, p. 2022.

[11] M. J. Gierl and H. Lai, "The Role of Item Models in Automatic Item Generation," *Int J Test*, vol. 12, no. 3, pp. 273–298, Jul. 2012.

[12] A. A. B. Prasetyanto, T. B. Adji, and I. Hidayah, "Automatic Question Generator System Conceptual Model for Mathematic and Geometry Parallel Question Replication," *J Phys Conf Ser*, vol. 1577, no. 1, p. 012023, Jul. 2020.

[13] S. E. Prasetyo, T. B. Adji, and I. Hidayah, "Automated Item Generation: Model and Development Technique," in *2020 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, Sep. 2020, pp. 64–69.

[14] T. Bharata Adji, F. Setio Pribadi, H. Eko Prabowo, R. Rosnawati, and A. Wijaya, "Generating Parallel Mathematic Items Using Automatic Item Generation," in *ICEAP Proceeding Book Vol 1*, Dec. 2018, pp. 89–93.

[15] A. Kr. Pandey and S. Singh, "Derivatives: A Comprehensive Study of Rate of Change," *Journal of Applied Science and Education (JASE)*, vol. 1, no. 1, pp. 1–4, Nov. 2021.

[16] Y. A. Brychkov, *Handbook of Special Functions: Derivatives, Integrals, Series and Other Formulas*. Chapman and Hall/CRC, 2008.

[17] L. A. R. Laliyo, D. N. Botutihe, and C. Panigoro, "The Development of Two-Tier Instrument Based On Distractor to Assess Conceptual Understanding Level and Student Misconceptions in Explaining Redox Reactions," *International Journal of Learning, Teaching and Educational Research*, vol. 18, no. 9, pp. 216–237, Sep. 2019.

[18] A. Meurer *et al.*, "SymPy: symbolic computing in Python," *PeerJ Comput Sci*, vol. 3, p. e103, Jan. 2017.

[19] F. Stoehr *et al.*, "Natural language processing for automatic evaluation of free-text answers — a feasibility study based on the European Diploma in Radiology examination," *Insights Imaging*, vol. 14, no. 1, p. 150, Sep. 2023.

[20] G. Lukács and A. Gartus, "Precise display time measurement in JavaScript for web-based experiments," *Behav Res Methods*, vol. 55, no. 3, pp. 1079–1093, May 2022.