UDK: 004.021:004.8:519.24:378

DEVELOPMENT OF AN INTELLIGENT ASSIGNMENT GENERATOR FOR STUDENTS BASED ON QUANTITATIVE MODELING

Ulashev Asrorjon Nasriddinovich
Associate Professor, Jizzakh Branch of the National University of Uzbekistan
Email: ulashevasror8@gmail.com
ORCID: 0000-0001-6049-8267

Annotatsiya. Ushbu tadqiqot masofaviy ta’lim platformalari uchun topshiriqlar murakkabligini talabalarning bilim darajasiga avtomatik moslashtiruvchi intellektual topshiriqlar generatorini ishlab chiqishga bag‘ishlangan. Tizim o‘quvchilar faoliyatini test natijalari, topshiriqlarni bajarish holati va faollik ko‘rsatkichlaridan olingan matematik asoslangan indikatorlar yordamida baholaydi hamda mos topshiriqlarni dinamik tarzda shakllantiradi. Qo‘llanilgan yondashuv adaptiv o‘qitish tamoyillariga asoslangan bo‘lib, talaba natijalaridagi o‘zgarishlarga mos ravishda topshiriqlarni uzluksiz yangilab borishni ta’minlaydi. Tajriba natijalari topshiriqlarni tayyorlashga sarflanadigan vaqtning sezilarli darajada qisqarganini va ta’lim natijalari yaxshilanganini ko‘rsatib, taklif etilgan modelning shaxsiylashtirilgan hamda samarali o‘quv jarayonlarini qo‘llab-quvvatlashdagi samaradorligini tasdiqlaydi.

Kalit so‘zlar: intellektual topshiriqlar generatori; masofaviy ta’lim; adaptiv o‘qitish; matematik modellashtirish; avtomatlashtirish; bilimlarni baholash.

Аннотация. В данном исследовании представлена разработка интеллектуального генератора заданий для платформ дистанционного обучения, который автоматически адаптирует уровень сложности заданий к уровню знаний обучающихся. Система оценивает учебную деятельность студентов с использованием математически обоснованных показателей, полученных на основе результатов тестирования, выполнения заданий и показателей активности, и динамически формирует соответствующие задания. Предложенный подход основан на принципах адаптивного обучения и обеспечивает непрерывное обновление заданий в ответ на изменения в учебных результатах студентов. Экспериментальные результаты показывают существенное сокращение времени, затрачиваемого на подготовку заданий, а также улучшение учебных результатов, что подтверждает эффективность предложенной модели для поддержки персонализированных и эффективных образовательных процессов.

Ключевые слова. интеллектуальный генератор заданий; дистанционное образование; адаптивное обучение; математическое моделирование; автоматизация; оценка знаний.

ISSN 3030-3907

Abstract. This study presents the development of an intelligent assignment generator for distance learning platforms that automatically adapts task difficulty to students' knowledge levels. The system evaluates learner performance using mathematically grounded indicators derived from test results, assignment completion, and activity metrics, and dynamically generates appropriate assignments. The applied approach is based on adaptive learning principles and enables continuous updating of tasks in response to changes in student performance. Experimental results indicate a notable reduction in assignment preparation time and improved learning outcomes, demonstrating the effectiveness of the proposed framework for supporting personalized and efficient learning processes.

Keywords: intelligent assignment generator; distance education; adaptive learning; mathematical modeling; automation; knowledge assessment.

INTRODUCTION

The rapid digitalization of education has led to the widespread adoption of distance and blended learning models. While online platforms provide flexibility and accessibility, one of the major challenges remains the effective assessment of students' knowledge and skills. Traditional assignment systems often rely on static and manually prepared tasks, which do not adequately reflect individual differences in students' learning progress [1].

Previous studies have underscored the importance of adaptive learning and personalized assessment. However, the integration of mathematical modeling with automated assignment generation remains insufficiently explored. This research addresses this gap by developing an intelligent assignment generator that dynamically adapts task difficulty based on quantitative knowledge assessment [2].

The main objective of this study is to develop an automated and adaptive assignment generation system and to evaluate its effectiveness using mathematical models. The research contributes a universal framework applicable to various distance learning platforms.

METHODS

System Architecture

The proposed system follows a modular architecture consisting of a database layer, a knowledge assessment module, an assignment generation module, an evaluation module, and a user interface. This structure ensures scalability and flexibility [5]. The modular design allows each component to function independently while exchanging necessary data, which simplifies integration with existing learning platforms.

Mathematical Model of Knowledge Assessment

ISSN 3030-3907

A students knowledge level is quantified using a weighted performance model (Equation 1). In this model, the knowledge level is calculated as a weighted combination of the students test score, assignment performance, and learning activity indicator:

$$K_i = w_1 T_i + w_2 A_i + w_3 L_i, \text{where } \sum_{k=1}^{3} w_k = 1 \qquad (1)$$

where

- K_i denotes the overall knowledge level of the i-th student,
- T_i represents the normalized test score,
- A_i denotes the normalized assignment performance measure,
- L_i represents the normalized learning activity indicator, and
- w_1, w_2, w_3 are non-negative weighting coefficients reflecting the relative importance of each component.

The weighting coefficients can be adjusted to emphasize specific performance aspects in accordance with pedagogical priorities and instructional objectives.

Based on the calculated knowledge level, the system selects an appropriate assignment difficulty using a mapping function (Equation 1). For example, the students knowledge score can be mapped to discrete difficulty levels (e.g., *easy*, *medium*, *hard*) by defining threshold ranges. This ensures that each student receives assignments of a difficulty commensurate with their current understanding.

After each assessment cycle, the knowledge level is updated to reflect the students new performance results. The update uses an adaptive formula (Equation 2) that incorporates the latest performance outcome. For instance, one can use an exponential smoothing or weighted update rule:

$$K_i^{\text{new}} = (1 - \alpha)\, K_i^{\text{old}} + \alpha R, 0 < \alpha < 1 \qquad (2)$$

Where R denotes the current performance result (e.g., the score obtained on a newly completed assignment or test), and $\alpha$is the adaptation coefficient (learning rate).

This adaptive knowledge update formula smooths short-term fluctuations by balancing historical performance $\left(K_i^{\text{old}}\right)$ with the most recent result. Smaller values of αassign greater weight to past performance, resulting in slower adaptation, whereas larger values of αincrease sensitivity to recent performance changes.

ASSIGNMENT GENERATION ALGORITHM

Using the above models, an assignment generation algorithm dynamically creates and assigns tasks suited to each students level. The algorithm consists of the following steps:

ISSN 3030-3907

1. Collect student performance data: Gather the students recent performance metrics, including test scores, assignment results, and activity levels.
2. Compute the knowledge level: The students current knowledge level K_i is computed by applying the weighted knowledge assessment model defined in Equation (1) to the collected performance indicators.
3. Determine assignment difficulty: The assignment difficulty level is selected by mapping the computed knowledge score K_i to discrete difficulty classes (easy, medium, hard) using the threshold-based mapping function defined in Equation (2).
4. Select or generate tasks: From the assignment database, select (or algorithmically generate) tasks of the corresponding difficulty level. A pool of tasks is maintained for each difficulty category to enable variety and relevance.
5. Deliver assignments: Present the selected tasks to the student through the learning platforms interface.
6. Evaluate results and update knowledge: Upon task completion, the students performance outcomes are assessed, and the corresponding knowledge level $K_i$is adaptively updated according to Equation (2), ensuring a smooth integration of recent performance into the historical knowledge estimate.
7. Repeat adaptively: Repeat the cycle for subsequent assignments, continuously adapting task difficulty as the students performance evolves.

This algorithm ensures an *adaptive learning loop* where each students progress guides the difficulty of future assignments. By continuously aligning task complexity with the learners current capabilities, the system avoids both overly trivial tasks for advanced students and overly challenging tasks for those struggling, thereby maintaining optimal engagement and learning efficacy.

RESULTS

Adaptive Dashboard Metrics and Progress Model

To evaluate and illustrate the systems behavior, a student dashboard was developed to track learning progress and activity. The dashboard displays several key indicators, each underpinned by a mathematical model:

- **Weekly learning progress.** Displays the proportion of the assigned weekly workload that has been completed by the student, expressed as a percentage (e.g., *"70% completed this week"*).
- **Subject-wise progress.** Displays the assignment completion status for each subject, typically visualized using individual progress bars for each course.
- **Daily task completion.** Displays the students completion status of tasks scheduled for the current day, providing a snapshot of short-term learning activity.

ISSN 3030-3907

- **Activity feed.** Records recent student interactions, such as assignment submissions and received feedback, and provides input data for the calculation of the activity index.

•**Recommended lessons.** Provides targeted learning resources for subjects where the student is lagging behind, supporting timely intervention and adaptive learning. These elements are modeled quantitatively as described below.

For each subject j, the subject-level assignment progress is defined as the ratio of completed assignments to the total number of assigned tasks in that subject. Formally, the progress for subject $j$is given by

$$P_j = \frac{C_j}{N_j}, 0 \le P_j \le 1, \qquad (3)$$

Where N_j denotes the total number of assignments in subject j, and $C_j$represents the number of assignments successfully completed by the student. This metric yields a normalized value between 0 and 1 (or equivalently 0–100%), indicating the students progress within the given subject.

For example, if a student completes 8 out of 10 assignments in *Mathematics*, the corresponding progress value is $P_{\text{Mathematics}} = 0.8(80\%)$. Accordingly, the dashboard visualizes this information using a progress indicator (e.g., *"Mathematics – 80% complete"*).

To obtain an overall measure of the students weekly progress across all subjects, a weighted progress metric is employed, which typically corresponds to the predefined weekly completion target. Since individual subjects may differ in workload or academic importance, a weighted aggregation of subject-level progress is applied. The overall progress is computed as

$$P_{\text{overall}} = \frac{\sum_j w_j \ C_j}{\sum_j w_j \ N_j}, \sum_j w_j = 1, \qquad (4)$$

where w_j denotes the weight associated with subject j, reflecting its relative importance or credit value.

This weighted ratio ensures that subject priorities are appropriately accounted for when computing total learning progress. To align the resulting value with the weekly completion target displayed on the dashboard, the normalized progress is converted into a percentage form, as defined in Equation (4). In practical terms, this involves mapping the fractional value of P_{overall} to a percentage scale and comparing it against a predefined weekly goal.

For instance, if the weekly target is set to 70% and the computed value satisfies $P_{\text{overall}} = 0.70$, the dashboard displays *"70% of weekly goal achieved."*

ISSN 3030-3907

Values below the target indicate insufficient progress, whereas values exceeding 1.0 may correspond to advanced completion or extra-credit activity. This behavior directly corresponds to the *"70% completed"* weekly progress indicator illustrated in the example dashboard.

The daily task completion rate is computed analogously to the subject-level progress metric. Each task scheduled for the current day is modeled as a binary completion variable. For task $i$assigned on a given day, the completion indicator is defined as

$$d_i = \begin{cases} 1, & \text{if task } i \text{ is completed,} \\ 0, & \text{if task } i \text{ is not completed.} \end{cases} \qquad (5)$$

If a total of Dtasks are assigned for the current day, the daily completion rate is calculated as the ratio of completed tasks to the total number of assigned tasks:

$$\delta_{\text{today}} = \frac{1}{D}\sum_{i=1}^{D} d_i \,, 0 \le \delta_{\text{today}} \le 1. \qquad (6)$$

This normalized value represents the proportion of daily tasks successfully completed and is typically displayed on the dashboard as a percentage of *"todays tasks"* finished.

Interpretation. A value of $\delta_{\text{today}} = 0(0\%)$ indicates that no tasks have been completed on the current day, whereas $\delta_{\text{today}} = 1(100\%)$ signifies full completion of all assigned daily tasks.

In addition to progress-based indicators, an overall engagement metric—referred to as the *activity index*—is computed to capture the students learning behavior in a more comprehensive manner (as illustrated in the *Activities* panel of the dashboard). This activity index is formulated as a multi-criteria composite measure that integrates several dimensions of learner behavior, including consistency, recency, and activity volume.

Formally, the activity index is expressed as a weighted combination of its constituent components:

$$I_{\text{activity}} = f(S, R, V), \qquad (7)$$

Where S denotes the activity streak (i.e., the number of consecutive active days), Rrepresents a recency factor reflecting how recently the student was active, and V corresponds to the activity volume, measured as the number of tasks completed within a specified time window (e.g., the previous seven days). In practice, each component is first normalized prior to aggregation.

- **Streak normalization.** The activity streak component is normalized as

$$S' = \frac{S}{S_{\max}}, 0 \le S' \le 1, \qquad (8)$$

ISSN 3030-3907

where S denotes the number of consecutive active days, and $S_{\max}$ is a predefined maximum streak length introduced to cap excessively long streaks and to prevent their disproportionate influence on the overall activity index.

•**Recency decay.** The recency component is modeled using an exponential decay function:

$$R' = e^{-\lambda \Delta t}, 0 \leq R' \leq 1, \qquad (9)$$

where Δt denotes the number of days since the students most recent activity, and $\lambda > 0$ is a decay coefficient controlling the rate at which the activity score decreases over time. This formula ensures recent activity contributes more strongly to the overall activity index, while prolonged inactivity leads to a gradual decline in news rankings.

Activity volume normalization. The activity volume component is normalized as

$$V' = \frac{V}{V_{\max}}, 0 \leq V' \leq 1, \qquad (10)$$

where V denotes the number of tasks completed by the student within a fixed recent time window (e.g., the previous seven days), and $V_{\max}$ represents a predefined maximum workload used to normalize activity volume and to prevent excessive influence of unusually high task completion rates.

These normalized components are subsequently aggregated into a single engagement metric using a weighted combination, subject to appropriate normalization constraints (e.g., the sum of weights equals one). Formally, the composite activity index is defined as

$$I_{\text{activity}} = \lambda_1 S' + \lambda_2 R' + \lambda_3 V', \sum_{k=1}^{3} \lambda_k = 1, \qquad (11)$$

where S', R', and V'denote the normalized streak, recency, and activity volume components, respectively, and λ_k are non-negative weighting coefficients reflecting their relative importance.

The resulting composite index I_{activity} takes values in the interval $[0, 1]$ (or equivalently 0–100%) and represents the students overall engagement level.

Interpretation. For example, a value of $I_{\text{activity}} = 0.56(56\%)$ indicates a moderate and stable level of engagement, suggesting that the student demonstrates consistent activity while still having potential for increased regularity or workload intensity.

The system further incorporates a recommendation engine designed to suggest supplementary learning materials based on the students current progress. A lesson is recommended when the students progress in a given subject falls below

ISSN 3030-3907

a predefined threshold, as formalized in Equation (11).

Specifically, let θ denote a minimum acceptable progress level (e.g., $\theta = 0.5$, corresponding to 50% completion). If the progress for subject j satisfies

$$P_j < \theta, \qquad (12)$$

The system automatically generates and displays additional recommended lessons or resources for that subject in the *"Lessons for you"* section of the dashboard.

This threshold-based mechanism explains the presence of recommended lessons for subjects such as *English* and *Mathematics* in the sample interface, as their corresponding progress indicators were below the predefined target level, thereby triggering the recommendation process.

Finally, all these computations are brought together in the dashboard via a progress engine algorithm that runs each time the dashboard is refreshed or the student completes a task. The computation flow for updating the students dashboard is as follows:

Algorithm: Student Dashboard Progress Engine

1. **Data retrieval:** All subjects associated with the student, along with the most recent assignment data—including assigned tasks, completion status, and obtained scores—are retrieved from the system database**.**

2. **Subject-level progress computation:** For each subject j, the subject-level progress P_j is computed in accordance with Equation (3).

3. **Overall progress computation:** Subject-level progress values are aggregated using predefined weighting coefficients to calculate the overall progress measure P_{overall}, as defined in Equation (4).

4. **Progress normalization:** The overall progress measure P_{overall} is mapped onto a percentage scale by normalization against the predefined weekly target, resulting in the weekly completion rate specified in Equation (4).

5. **Daily task evaluation:** The completion status of tasks assigned for the current day is assessed to compute the daily completion rate δ_{today}, as defined in Equation (6).

6. **Activity index computation:** The students current engagement is evaluated by calculating the composite activity index I_{activity}, which integrates normalized streak, recency, and activity volume components as specified in Equation (11) and subsequent sub-formulas.

7. **Recommendation generation:** The system evaluates subject-level progress and identifies subjects with $P_j < \theta$, subsequently generating targeted learning recommendations for those subjects as specified by Equation (12).

ISSN 3030-3907

8. **Dashboard update:** All dashboard components are refreshed to reflect the latest computed values, including subject-level progress bars, overall weekly completion indicators, daily task completion status, the activity engagement index, and recommended lessons.

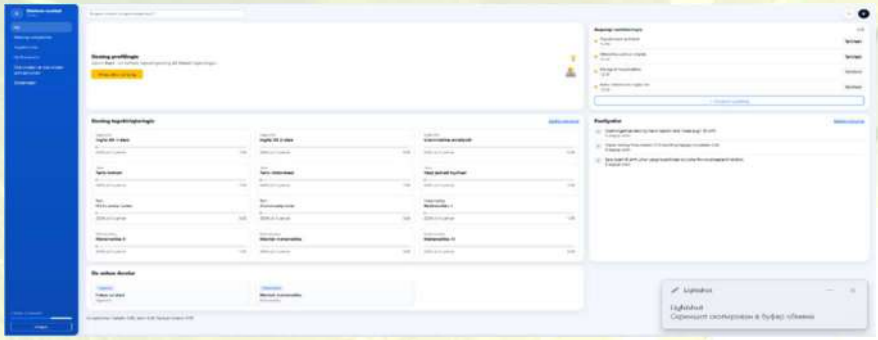


Figure 1. Adaptive assignment management and analytics dashboard.

Each of the main dashboard indicators is based on a specific type of algorithm, as summarized below:

Table-1.

Algorithmic aggregation of key performance indicators

Indicator	**Algorithm Used**	**Mathematical Basis**
Weekly Progress	Weighted ratio algorithm	Deterministic (linear)
Today's Completion	Conditional ratio algorithm	Deterministic (linear)
Bayesian Performance (β_{total})	Beta-Binomial Bayesian inference	Probabilistic (Bayesian)
Activity Index	Normalized composite index	Multi-criteria (hybrid)



Figure 2. Learner analytics dashboard based on deterministic and Bayesian performance indicators.

Bayesian Performance Estimation (β_{total})

A key contribution of the proposed dashboard is the introduction of a Bayesian performance coefficient, β_{total}, which yields a statistically grounded estimate of student task success by incorporating uncertainty associated with limited observational data. The coefficient is formulated within a Beta–Binomial Bayesian modeling framework:

1. Each task outcome is modeled as a Bernoulli random variable

$$X_i \sim \text{Bernoulli}(p),$$

ISSN 3030-3907

where $p$denotes the students probability of successful task completion.

2. **Prior distribution (uncertainty modeling).** Prior to incorporating the students observed performance data, a prior probability distribution is specified for the success probability p. A conjugate Beta prior is adopted:

$$p \sim \text{Beta}(\alpha_0, \beta_0), \qquad (13)$$

where $\alpha_0 > 0$ and $\beta_0 > 0$ are hyperparameters encoding prior beliefs about the students task success rate. In the absence of prior knowledge, a non-informative choice $\alpha_0 = \beta_0 = 1$ is employed, which corresponds to a uniform distribution over the interval $[0, 1]$.

The use of a Beta prior ensures statistical stability of the estimation process, particularly in scenarios with a limited number of observations, by preventing extreme probability estimates at early learning stages.

3. **Observation aggregation.** After the student completes N assignments (trials), the total number of successful task completions is denoted by

$$K = \sum_{i=1}^{N} X_i , \qquad (14)$$

where $X_i \in \{0,1\}$ represents the outcome of the i-th task. This aggregation summarizes the cumulative evidence obtained from all observed task outcomes and serves as the sufficient statistic for subsequent Bayesian updating.

4. **Posterior distribution (Bayesian update).** By applying Bayes' theorem, the prior distribution is updated using the observed task outcomes. Owing to the conjugacy between the Bernoulli likelihood and the Beta prior, the posterior distribution of the success probability $p$is obtained in closed form as

$$p \mid \mathbf{X} \sim \text{Beta}(\alpha_0 + K,\ \beta_0 + N - K), \qquad (15)$$

where K denotes the number of successful task completions and N is the total number of observed tasks. This posterior distribution represents the updated belief about the students underlying probability of successful task completion after incorporating evidence from N observed assignments.

5. **Expected value (dashboard output).** The Bayesian performance coefficient β_{total} is defined as the expected value of the posterior distribution of the success probability p. For a Beta-distributed posterior, this expectation admits a closed-form expression. Specifically, given

$$p \mid \mathbf{X} \sim \text{Beta}(\alpha_0 + K,\ \beta_0 + N - K),$$

the expected value is computed as

$$\beta_{\text{total}} = \mathbb{E}[\, p \mid \mathbf{X}\,] = \frac{\alpha_0 + K}{\alpha_0 + \beta_0 + N}. \qquad (16)$$

ISSN 3030-3907

This coefficient takes values in the interval [0, 1] (or equivalently 0–100%) and serves as the Bayesian performance indicator displayed on the dashboard.

$$\beta_{\text{total}} = \frac{\alpha_0 + K}{\alpha_0 + \beta_0 + N}. \qquad (17)$$

The coefficient β_{total} is displayed on the dashboard as the *Bayesian performance* indicator. It represents a statistically adjusted estimate of the probability of successful task completion, obtained by integrating the students observed performance outcomes with prior belief encoded by the Beta distribution. This Bayesian formulation provides a more stable and reliable performance measure than raw completion ratios, particularly in scenarios with limited observational data.

Interpretation. For example, a value of $\beta_{\text{total}} = 0.48(48\%)$ indicates that, given the students observed performance data and the assumed prior distribution, the model estimates an approximately 48% probability of successfully completing a newly assigned task. Compared to a raw success rate, this Bayesian estimate provides a more stable and reliable performance indicator, particularly when the number of observed tasks N is small, as the Bayesian framework naturally shrinks extreme estimates toward the prior expectation.

Recent Submissions Ordering and Scoring Model

Another component of the system is the *Recent Submissions* panel, which displays the students most recent assignment submissions together with their corresponding scores. The ordering and scoring of submissions in this panel follow a predefined algorithmic principle.

Let S denote the set of recent submissions associated with a given student:

$$S = \{s_1, s_2, \dots, s_n\}. \qquad (18)$$

Each submission $s \in S$ is represented as an ordered tuple:

$$s = (\text{course}(s), \text{date}(s), \text{score}(s)), \qquad (19)$$

Where $\text{course}(s)$ denotes the course or subject name, $\text{date}(s)$ represents the submission timestamp, and $\text{score}(s)$ is the normalized performance score obtained for that submission.

The set of submissions is ordered using a composite sorting key that prioritizes chronological recency and performance quality, as formalized in Equation (19).

This ordering strategy ensures that the *Recent Submissions* panel simultaneously highlights the most up-to-date student activity and draws attention to assignments with stronger performance outcomes.

Scoring Normalization Model

ISSN 3030-3907

Each submission score is computed using a normalization procedure that relates the raw points earned by the student to the maximum possible score for the corresponding task. Formally, if a student obtains a raw score R_{obtained} out of a maximum attainable score R_{max}, the normalized score is defined as

$$S_{\text{norm}} = \frac{R_{\text{obtained}}}{R_{\text{max}}}, 0 \leq S_{\text{norm}} \leq 1. \qquad (20)$$

This normalization yields a dimensionless performance measure in the interval $[0, 1]$, which may equivalently be expressed as a percentage (0–100%) for visualization purposes.

To ensure informative and interpretable scoring, the following special cases are considered:

- If a task is completed with a perfect score, the normalized score is set to $S_{\text{norm}} = 1(100\%)$.
- If a task is partially completed or completed with penalties (e.g., late submission), the normalized score assumes a fractional value in the interval $(0, 1)$, reflecting partial credit.
- If a task is not completed or the student fails the task, the normalized score is set to $S_{\text{norm}} = 0$.

Aggregation Logic for Analytics Modules: Normalized submission scores are used by higher-level analytics modules to compute aggregate indicators such as weekly learning progress and activity engagement. These scores are integrated through an aggregation mechanism (Equation 20) based on weighted summation or averaging, producing consolidated performance measures for analytical evaluation. In addition, the Bayesian performance coefficient β_total aggregates binary success and failure outcomes within a probabilistic framework, complementing deterministic metrics with a statistically robust estimate of overall task success probability.

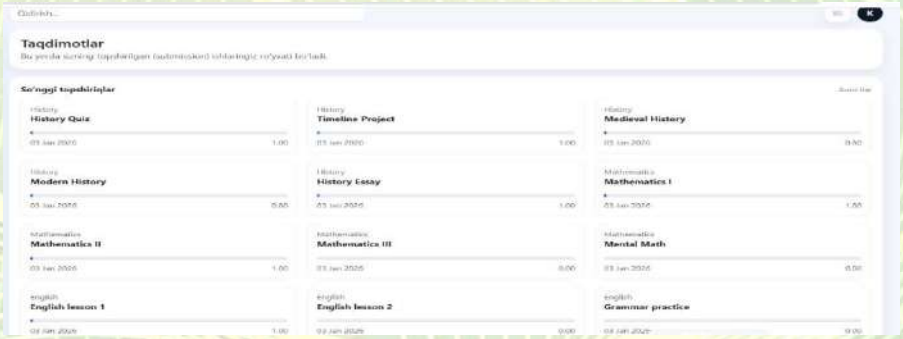


Figure 3. Recent submissions panel showing normalized scores ordered by recency.

For improved readability and interpretability, submissions are grouped by subject, enabling efficient scanning of recent learner activity. In addition, higher-

ISSN 3030-3907

scoring submissions may be visually emphasized through interface cues; however, chronological ordering remains the dominant criterion to highlight the recency of completed work.

Help and Resources Module Algorithms: The platform's Help and Resources module supports students through structured guidance and FAQs, using simple and explainable algorithms to ensure usability, transparency, and accessibility. It consists of three core components— a platform usage guide, a task submission guide, and an FAQ section—presented in a unified card-based interface that enhances visual clarity and enables intuitive navigation.

Platform Usage Guide. The platform usage guide provides introductory instructions for new users and functions as a beginner-oriented support component. Content delivery is controlled by a simple, deterministic rule-based algorithm that evaluates the user's experience level or account age.

If a user is identified as a beginner, such as during the first login or the initial period of platform usage, the guide is displayed; experienced users do not see this content. This approach ensures relevant onboarding support while maintaining transparency and avoiding unnecessary complexity.

Task Submission Guide: The task submission guide provides step-by-step instructions for uploading and submitting assignments through a sequential workflow based on a finite-state machine (FSM). Each step represents a distinct state, with deterministic transitions that require successful completion of the current step before proceeding. This FSM-based approach enforces correct submission procedures and reduces common errors, such as missing file attachments or skipped confirmation steps.

Frequently Asked Questions (FAQ): The FAQ module uses a keyword-based information retrieval algorithm to provide fast and transparent access to common support information. User queries are processed by extracting relevant keywords and matching them with predefined keyword sets for each FAQ entry. When a sufficient match is found, the corresponding answer is displayed, enabling immediate responses without relying on complex natural language processing or machine learning techniques**.**

Content Layout and Rendering. All support components—the platform usage guide, task submission instructions, and FAQ module—are displayed using a unified card-based grid layout. A dedicated layout algorithm ensures consistent sizing, visual style, and alignment of help cards, supporting clarity and responsive design.

The algorithm dynamically arranges only active components (e.g., hiding the usage guide for non-beginner users) and evenly distributes them within the grid to

ISSN 3030-3907

maintain visual balance and usability. This adaptive layout results in a clean and accessible *Help and Resources* section.

Table 2.

General information about algorithms in the Help and Resources module

Section	Purpose	Applied Algorithm
Platform Usage Guide	Introductory guidance	Rule-based content selection
Task Submission Guide	Procedural support	Finite-state workflow (FSM)
FAQ	Quick assistance	Keyword-based retrieval
Layout	Visual organization	Grid layout rendering



Figure 4. Help and Resources panel based on deterministic and explainable support algorithms.

Experimental Evaluation: The proposed system was evaluated in a real distance learning environment to assess its effectiveness. Results indicate that the intelligent assignment generator significantly streamlines course management, reducing instructors' time spent on preparing personalized assignments by approximately 60–70%. Students' average performance increased by 30–45% following the introduction of adaptive assignments, with the greatest improvements observed in analytical and problem-solving tasks. Variance analysis revealed more consistent learning outcomes across student groups, as weaker students improved while stronger students maintained high performance. These findings highlight the efficiency and educational benefits of an adaptive, data-driven assignment generation approach.

DISCUSSION

The results demonstrate that the proposed automated and adaptive assignment generation framework effectively improves learning efficiency by continuously aligning task difficulty with individual learner performance. By integrating mathematically grounded indicators—weighted progress ratios, Bayesian performance estimation (β_total), and a composite activity index—the system provides a stable and objective assessment of student engagement and achievement.

Compared to traditional static assignment systems, the proposed approach dynamically adapts learning content using real-time progress and historical

ISSN 3030-3907

performance data. This ensures that advanced learners are not limited by repetitive tasks, while struggling learners receive focused reinforcement on core concepts. Notably, the Bayesian performance model reduces uncertainty arising from sparse data and delivers reliable estimates even in the early stages of learning.

The modular architecture further enhances the framework by enabling seamless integration with existing learning management systems without major structural modifications. Well-defined interfaces between independent analytical components support scalability, simplify maintenance, and promote transparency. The deterministic and explainable nature of the algorithms fosters trust and interpretability, which are essential for the effective adoption of intelligent learning technologies in educational environments.

CONCLUSION

This study introduced an intelligent assignment generation and evaluation system based on mathematical modeling and adaptive learning principles. The framework automates the entire assignment lifecycle: it generates assignments tailored to each student, quantitatively evaluates learner performance on those assignments, and dynamically adjusts subsequent task difficulty based on measurable indicators of progress and activity. The integration of ratio-based performance metrics, Bayesian inference for success estimation, and composite indexing of engagement provides a robust and scalable solution for both distance and blended learning environments.

The proposed system enhances learning outcomes by personalizing educational content to each students needs, while simultaneously reducing the instructors' workload through automation of routine task preparation and grading analysis. Its algorithmic transparency and modular design make the framework universal and easily adaptable to various educational platforms and subjects. In the future, this work can be extended by incorporating more advanced artificial intelligence techniques, such as machine learning-based prediction models for student performance and content recommendation. Additionally, deeper learning analytics (for example, analyzing solution steps or common errors) could be integrated to further improve the systems adaptability and the accuracy of its interventions. Such enhancements could lead to even more effective intelligent tutoring systems and adaptive learning platforms.

REFERENCES

1. Woolf, B. P. *Building Intelligent Interactive Tutors*. San Francisco: Morgan Kaufmann, 2010.

ISSN 3030-3907

2. Bishop, C. M. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.
3. Murphy, K. P. *Machine Learning: A Probabilistic Perspective*. Cambridge: MIT Press, 2012.
4. Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. *Bayesian Data Analysis* (3rd ed.). Boca Raton: CRC Press, 2013.
5. Ulashev, A., & Aynakulov, T. Saralash algoritmlarining didaktik ahamiyati va o‘quv jarayonidagi roli. Proceedings of Scientific-Practical Conference, 2025, Vol. 1, No. 1, pp. 6–9. (In Uzbek)
6. Nasriddinovich, U. A., et al. Mobil ta’lim ilovalari orqali intellektual metodlarni rivojlantirish. *Global Science Review*, 2025, Vol. 1, No. 1, pp. 37–42. (In Uzbek)
7. Aynakulov, T., Mahkamov, S., & Ulashev, A. Tasvirlardan xususiyatlarni avtomatik ajratib olish, obyektlarni aniqlash va klassifikatsiya qilish. Proceedings of Scientific-Practical Conference, 2025, Vol. 1, No. 1, pp. 46–50. (In Uzbek)
8. Nasriddinovich, U. A. Intelektual simulyatsiya tizimlari yordamida saralash algoritmlarini o‘rgatish metodikasi. *Global Science Review*, 2025, Vol. 1, No. 1, pp. 31–36. (In Uzbek)
9. Ulashev, A. N. Intellektual simulyatsiya tizimlari orqali talabalarning saralash algoritmlari bo‘yicha kompetensiyalarini rivojlantirish metodologiyasi. *Ekonomika i Sotsium* (Economics and Society), 2025, No. 2-1(129), pp. 636–639. (In Uzbek/Russian)
10. Nasriddinovich, U. A., et al. C# da web kamera orqali QR kod yasash va o‘qish texnologiyasi. *Journal of Integrated Education and Research*, 2023, Vol. 2, No. 4, pp. 30–34. (In Uzbek)
11. Mamasolievich, J. D., & Nasriddinovich, U. A. *Matematika darslarida dasturlashtirilgan ta'lim* (Programmed Learning in Mathematics Classes). *Nauchnyy Fokus* (Scientific Focus), 2024, 1(12), pp. 384–388.
12. Nasriddinovich, U. A. (2024). Axborotlashgan pedagogik texnologiyalar–ta’lim samaradorligini oshirishning samarali vositasi. *Tadqiqotlar. UZ*, *37*(3), 166-168.
13. Nasriddinovich U. A., Xayriddin o‘g‘li E. F. Elektron fitness klubining dasturiy taminotini yaratish uslublari //Journal of new century innovations. – 2023. – T. 26. – C. 143-149.

ISSN 3030-3907