ingegneria sismica
International Journal of Earthquake Engineering
Anno XLIII - Num. 2 - 2026

A Tutorial Model based on Dynamically Generated Artificial Intelligence Aims to Enhance the Digital Media Learning Experience in Universities

Peng Wang[1] *and Huangfang Wan*[2,*]

[1] School of Literature, Journalism and Communication, Hubei Engineering University, Xiaogan 432000, Hubei, China

[2] Department of English and Law (General Education Department)，College of Technology, Hubei Engineering University, Xiaogan 432000, Hubei, China

SUMMARY: *Dynamic generative AI can support digital media learning by generating task-aware tutorials that are adapted to learner behavior, project status, and feedback trajectories. This paper proposes a digital media teaching model DGAI-Tutor. The system integrates software operation logs, project version records, prompt histories, work metadata, and peer feedback from 128 students over 12 weeks to form 6420 annotated learning segments. After data anonymization, timestamp alignment, feature normalization, semantic embedding and time window segmentation, a retrieval augmented large language model is combined with a Transformer learning state encoder and a path adapter. The model generates editorial guidance, concept explanations, and next step tasks for image design, video production, and interactive media projects. Experimental results show that DGAI-Tutor achieves 92.8% tutorial relevance, 89.6% path matching rate and 0.214 feedback MAE, and the 95-quartile delay of three types of tasks is less than 126ms. It is superior to rule retrieval generation and static tutorial recommendation in terms of adaptive guidance and learning experience prediction. The framework can be deployed in the classroom through lightweight caching and controlled generation modules.*

KEYWORDS: *Generative AI; Tutorial model; Digital media; Path adaptation*

1 Introduction

The learning process of digital media courses in universities relies on continuous creative tasks such as image processing, video editing, and 3D modeling. Students form process traces in software operation, material selection, parameter adjustment and work iteration. Traditional tutorials are mainly based on fixed text, unified video and after-class comments, which is difficult to generate executable step instructions according to the status of works. Dynamic generative AI can connect task semantics, operational context and learning feedback into the same computing link, making the tutorial content transform from static resources to a generation service that can be perceived, searchable and controlled. Alier et al. [1] have studied the transformation of generative artificial intelligence from an auxiliary tool to a learning infrastructure in educational Settings, showing that the content generation mechanism needs to establish a stable mapping with real learning tasks. Penalvo and Ingelmo[2] systematically composes the technical evolution, trend, and method structure of GenAI, which provides the conceptual foundation for large language models, diffusive

*binbinbin88@163.com
https://doi.org/10.65102/is2026965 Published: 30 April, 2026

content generation, and prompt control. Kasneci et al. [3] analyzed the opportunities and challenges of large language models in education, and pointed out that the generation of feedback needs to take into account accuracy, interpretability and learner differences.

Ray[4] made a comprehensive review of ChatGPT's background, applications, biases, ethics, and limitations. His research suggested that generative tutorials do not only rely on open-ended answers, but also need knowledge boundaries, content verification, and risk filtering. Guettala et al. [5] proposed that generative artificial intelligence can promote adaptive and personalized learning, indicating that learning paths, content difficulty and feedback methods can be dynamically adjusted according to learning states. Minn[6] studied the application of AI-assisted knowledge evaluation technology in adaptive learning environment, and showed that learner ability estimation can provide computational basis for tutorial recommendation and task stratification. Southworth et al. [7] proposed an AI literacy model for university courses, emphasizing that artificial intelligence ability should be embedded in the structure of professional courses. Verma et al. [8] designed a tool for understanding student engagement based on video conferencing context, which provides a basis for behavior perception, participation status recognition and feedback trigger in digital media learning. Hossen and Uddin[9] proposed an attention monitoring method for online classroom based on XGBoost, and proved that the interpretable classification model could identify state changes from behavioral features.

In university digital media learning, tutorial generation faces technical constraints. Image composition tasks focus on layer structures, color relationships, and export specifications. Video tasks involve timelines, shot editing, transition rhythms, and audio-visual synchronization. It is difficult for a single text question answering to understand these cross-modal processes, and it is difficult for static recommendation algorithms to maintain continuous guidance in the process of work modification. In this paper, we propose DGAI-Tutor, a dynamic generative AI tutorial model, to construct learning event encoding, knowledge retrieval, generation control, and path adaptation frameworks. The system collects software operation logs, project version records, prompt input and work metadata of 128 college students during their 12-week learning process, forming 6420 learning fragments. After semantic embedding, time window segmentation, task node encoding, and quality label alignment, the data enters the Transformer learning state encoder to generate the current ability, task progress, error type, and content requirement representation.

DGAI-Tutor consists of a retrieval enhancement generation module, a knowledge constraint layer, a tutorial path adapter, and a feedback evaluator. The retrieval enhancement module recalls relevant content according to the course knowledge graph, tool command library and work examples to reduce the drift of generated results. The knowledge constraint layer verifies the terms, step sequence, software parameters and work format, so that the generated tutorial can be operated accordingly. The path adapter selects the next task according to the learner state vector to avoid misalignment between tutorial difficulty and current ability. The feedback evaluator calculates learning experience metrics using text semantics, work version differences, and interaction delays, and outputs tutorial relevance, path matching rate, feedback error, and system response time.

The contributions are as follows: A dynamic tutorial generation model for digital media tasks is constructed to enable generative AI to understand the computational relationships between the work process, tool operations, and learning objectives. The joint mechanism of knowledge constraint and retrieval enhancement is designed to reduce the unfounded instructions and step jumps in the generated content. The learning state recognition and path adaptation algorithm is established, and the operation log, work iteration and text feedback are transformed into a computable tutorial adjustment basis. The goal is to form a deployable

intelligent tutorial system that can complete tutorial generation, immediate feedback, and learning experience prediction in image design, video production, and interactive media projects, and verify the model performance through indicators such as tutorial relevance, path matching rate, feedback MAE, and delay.

2 Related work

2.1 Application of Generative Artificial Intelligence in Intelligent tutorial System

After generative artificial intelligence enters the intelligent tutorial system, the tutorial content is no longer only dependent on preset courseware and item bank retrieval, but is driven by learning tasks, contextual semantics, knowledge constraints and user feedback. Dakakni and Safa[10] studied the ethics and fairness impact of artificial intelligence entering the language classroom in colleges and universities. Their analysis showed that the generative tutorial system needs to add identity desensitization, deviation detection and result audit modules before content output to avoid unstable results caused by model feedback due to training corpus or user differences.

Segbenya et al. [11] established an influence model around the relationship between the use intention of artificial intelligence in colleges and universities and employment skills, and proved that learners 'acceptance of AI tools would affect the frequency of system use and the depth of interaction. This conclusion can be transformed into a user state variable in the tutorial model, which can be used to adjust the granularity of the generated content and the way of prompting. Habibi et al. [12] studied the acceptance and use of ChatGPT in higher education learning, and pointed out that the conversational generation model can assume the functions of interpretation, example generation and task assistance, but the tutorial system still needs to record dialogue rounds, task nodes and feedback history to maintain a continuous learning context.

Albayati[13] analyzed the differences in undergraduates' perception of ChatGPT as a regular auxiliary tool from the perspective of user acceptance, and showed that the quality of prompts, output credibility and use experience would affect learners' dependence on the system. Existing research provides a basis for user behavior, system trust, and feedback interaction for generative artificial intelligence tutorial systems. However, most of the work still remains at the level of general question answering or learning attitude analysis, and lacks structured expressions for image editing, video synthesis, 3D modeling, and interactive scripting tasks in digital media courses.

The tutorial model for digital media learning needs to encode tool commands, works versions, material metadata and course knowledge graph together, so that the generated content can correspond to specific operations rather than general explanations. Therefore, the dynamic generative model should add retrieval enhancement, task constraint, path record and generation verification in addition to the large language model, so that the tutorial text, example code, operation steps and work feedback can form a traceable calculation process. In the implementation of the system, the generation end also needs to access the course resource index, work scoring rules and error sample library, and limit the output range through similar task recall. Then the software version, parameter name and step sequence are checked by the rule checkier, so that the tutorial content can directly serve the modification and iteration of digital media projects, and retain the generation evidence chain for subsequent review.

2.2 Digital media learning behavior modeling and multimodal data analysis

The learning behavior of digital media has the characteristics of continuous creation and cross-modal expression. The learning process usually leaves software operation logs, mouse tracks, layer modifications, timeline edits, material calls, text questions, version submissions and other data. Einarsson et al. [14] studied the application of ChatGPT in automatic problem reconstruction in different academic fields, and showed that the generative model could rewrite fuzzy requirements into tractable tasks. This idea can be used for creative intention analysis in digital media learning, and expressions such as "picture incongruity" and "editing influency" can be converted into computable labels such as color, composition, rhythm and interaction logic.

Al Shloul et al. [15] discussed the role of active-based learning and ChatGPT on student performance, indicating that AI feedback in task-driven environments needs to be embedded in specific activity processes. In the digital media course, the model should identify whether the student is in the stage of material sorting, effect debugging, work synthesis or export inspection, and match the corresponding tutorial. Alnasyan et al. [16] systematically analyzed the ability of deep learning technology to predict student performance in virtual learning environment, and pointed out that sequence model and feature fusion method can extract learning trends from platform records. This conclusion is suitable to be translated into operation sequence coding and work evolution prediction.

Baek et al. [17] studied the way and perception differences of college students' use of generative AI, revealing that students' use paths in creation assistance, explanation query and scheme comparison are not consistent. The above studies illustrate that digital media learning behavior cannot be counted only by the number of visits or assignment scores, but the state of creation should be recovered from multimodal evidence. The system can extract image works into color histogram, texture feature and layout structure, encode video works into shot length, transition type and audio-visual synchronization index, and convert interactive media items into event nodes, script calls and page jump graphs.

Text feedback and prompt input represent the learning goal, confusion type and modification intention through semantic embedding. After time alignment, the multi-source features are input into Transformer or graph neural network to form the learner state vector, which provides a data basis for subsequent tutorial generation, path adaptation and experience prediction. At the modeling level, continuous operations can be segmented by session Windows, changes of works can be represented by version difference, and interaction events can be transformed into directed behavior graphs. Different modalities are not required to be forcibly merged into a single feature, but to retain the source label, confidence and missing state under a unified timestamp, which is convenient for the model to judge the tutorial generation basis and maintain the traceability of data link.

2.3 Adaptive learning path recommendation and learning experience prediction methods

Adaptive learning path recommendation needs to represent the learner's current state, task difficulty, knowledge dependence and system feedback as a computable sequence. Baig and Yadegaridehkordi[18] studied the application literature of ChatGPT in higher education and pointed out that generative AI has been extended from general question answering to learning support, task explanation and feedback generation, but existing path recommendation often lacks fine-grained modeling of professional task processes.

Farrokhnia et al. [19] used SWOT method to analyze ChatGPT's impact on educational

practice and research, and proposed that the generative model has the ability of immediate feedback and content expansion, while maintaining output reliability, task boundaries and learning evidence. Cotton et al. [20] conducted a study on academic integrity in the ChatGPT era, indicating that automatically generated content must have traceable records and source constraints. In tutorial systems, this idea translates into generating logs, retrieving evidence, and version traces. Kamalov et al. [21] discussed the new stage of artificial intelligence education and proposed that AI systems are forming a multi-dimensional learning support structure, which provides a technical basis for personalized paths, automatic feedback and sustainable learning services.

A comparison of related studies is shown in Table 1. The table shows that the existing research has covered generative assistance, learning support, integrity constraints and intelligent education ecology, but the path recommendation for digital media learning still needs to further combine the work process and tool operation.

Table 1: Comparison of related studies on adaptive tutorial models

Research Source	Research Focus	Computational Method	Data or Scenario	Implications for the Proposed Model
Baig and Yadegaridehkordi [18]	Review of ChatGPT applications in higher education	Literature mapping and research challenge analysis	Higher education learning scenarios	Indicates that generative AI can be integrated into learning support, task explanation, and feedback generation modules
Farrokhnia et al. [19]	Strengths and limitations of ChatGPT in educational applications	SWOT analysis	Educational practice and research scenarios	Supports reliability control, task boundary constraints, and evidence recording in tutorial generation
Cotton et al. [20]	Academic integrity in the era of ChatGPT	Risk analysis and regulatory design	University learning and writing scenarios	Suggests that generated content should retain sources, versions, and usage traces
Kamalov et al. [21]	Development of artificial intelligence education systems	Multidimensional technical analysis	Intelligent learning and education systems	Provides systematic references for path recommendation, automatic feedback, and learning experience prediction

Graphic design, video production, and interactive media development are not linear learning-learners may revise materials, adjust parameters, go back to tutorials, and refactor solutions repeatedly in the same task. Therefore, the path recommendation algorithm should use the state transition method to describe the learning phase, and map each operation, each work version and each feedback text to the state node. The system can judge the prior skills according to the knowledge graph, estimate the current completion degree according to the work quality tags, and then use reinforcement learning or sequence recommendation model to select the next tutorial.

Learning experience prediction should not rely only on satisfaction questionnaires, but should incorporate indicators such as tutorial click stay, generated content adoption rate, task completion time, number of repeated revisions, and feedback error. With these metrics, the model is able to identify whether the generated tutorial is actually entering the authoring process. The dynamic tutorial system also needs to set a cold start mechanism on the recommendation side to initialize the learners who lack history records using course task templates and similar works samples. Individual state vectors are used to update recommendation weights for learners with existing records. After the generated content enters the interface, the system can continue to record whether the learner adopts the step, whether the operation is reversed, and whether the learner requests for explanation again, and write these data back to the path adaptation module. The closed loop thus formed is able to support the process instruction in the digital media course, rather than only giving evaluation after the assignment is submitted, and form a reviewable computational link.

3 Dynamic generative artificial intelligence tutorial model and digital media learning adaptation mechanism

3.1 Dynamic tutorial generation model for digital media learning tasks

Digital media learning tasks usually consist of target description, software operation, work status and feedback text. The dynamic tutorial generation model does not directly feed student questions into the big language model, but first encodes task semantics, project files, operation trajectories and course knowledge into a unified state, and then generates step-by-step tutorials according to the state. This design enables the content of the tutorial to correspond to specific tasks such as image retouching, video editing, and interactive page production, avoiding the generation of results remaining in concept explanation.

To illustrate the way of data flow inside the dynamic tutorial generation model, Fig. 1 shows the relationship between task input, semantic encoding, knowledge retrieval, content generation and output verification. After the task request enters the system, it first forms the task state together with the work version and operation log, and then recalls the course knowledge, tool commands and similar work cases through the retrieval enhancement module. The generation module outputs the tutorial steps under constraints, and the verification module checks the consistency of terms, parameters and operation sequence.

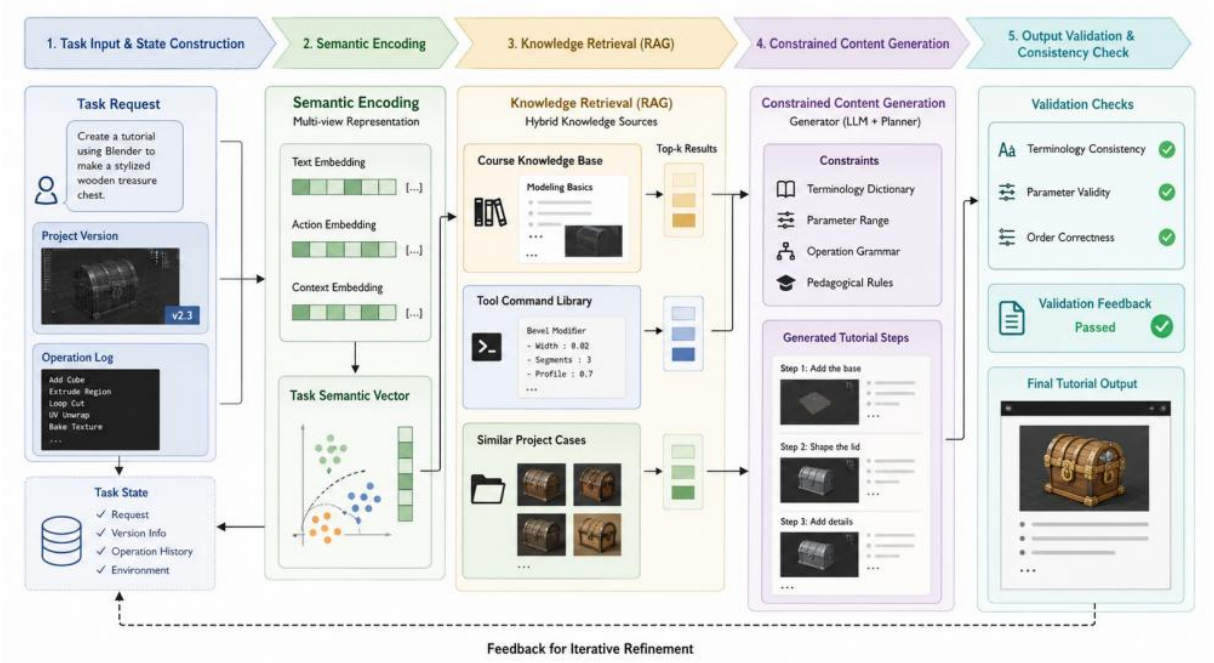

Figure 1: Task semantic encoding and tutorial output flow of dynamic tutorial generation model

To ensure that digital media tasks can be stably recognized by the model, the system compresses text requests, work metadata, operation sequences and knowledge nodes into the same semantic space, and retains the weight differences of different data sources. The task state vector is shown in the following equation:

$$z_t = \mathrm{LN}(W_q q_t + W_o o_t + W_a a_t + W_k k_t + b) \tag{1}$$

Here,z_t represents the task state vector of the t learning interaction. q_t represents the text requirements entered by the student. o_t represents work version metadata. a_t represents the software operation sequence; k_t represents the course knowledge node; W_q,W_o,W_a,W_k are trainable mapping matrices; LN denotes the layer normalization. The role of this formulation is to turn scattered learning evidence into a task representation that can be invoked by the generative model.

In order to reduce the content drift in tutorial generation, the retrieval enhancement module calculates the relevance of candidate knowledge fragments according to the task state vector, and introduces knowledge freshness and complexity penalties to make the recalled content not only fit the task, but also not exceed the current learning stage. The retrieval score is shown in the following equation:

$$s_j = \frac{z_t^{\top} r_j}{\sqrt{d}} + \alpha\rho_j - \beta c_j \tag{2}$$

where s_j denotes the recall score of the j knowledge fragment; r_j represents knowledge fragment embedding; d denotes the vector dimension; Let ρ_j denote the matching strength between the segment and the course task. c_j is the operation complexity. α and β are the adjustment coefficients. The formula can control the basis of tutorial generation and avoid the system referencing instructions unrelated to the work task.

In the tutorial generation stage, the model needs to simultaneously consider language

coherence, tool command executability and course specification constraints. The generation probability is determined by decoding hidden state, retrieving evidence and constraint mask. The output of tutorial word is shown in the following equation:

$$P(y_i|y_{<i}, z_t, R_t) = \text{Softmax}\left(\frac{W_h h_i + W_r \bar{r}_t + m_i}{\tau}\right) \tag{3}$$

where y_i represents the i generated lemma; $y_{<i}$ indicates a generated sequence. R_t represents the set of retrieved knowledge; h_i represents the decoder hidden state. $\bar{r}_t$ represents the aggregation vector of retrieval evidence. m_i represents the constraint mask; Let τ denote the temperature parameter. This formula makes the generation process constrained by task evidence and operation rules.

Digital media learning has the characteristics of repeated modification, and the tutorial model needs to update the learner's state with the change of works. The system will input the adoption behavior, question again, work difference and feedback score into the state update unit, and the learner state transition is shown in the following equation:

$$u_{t+1} = \text{GRU}(u_t, [e_t; \Delta o_t; f_t; p_t]) \tag{4}$$

where u_t represents the current learner state; e_t stands for tutorial interaction embedding;Δo_t represents the difference of works version; f_t represents the feedback rating vector. p_t stands for path node coding; GRU stands for Gated recurrent unit. This formula is used to characterize the continuous process of students from understanding, manipulation to modification.

In order to train a controllable and reviewable dynamic tutorial generation model, the objective function simultaneously constraints tutorial relevance, semantic consistency, path adaptation and parameter stability, so that the model output not only meets the language fluency, but also fits the work task. The comprehensive loss is shown in the following equation:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{ce} + \lambda_2 (1 - \cos(g, z_t)) + \lambda_3 \max(0, \delta - Q_t) + \lambda_4 \|\Theta\|_2^2 \tag{5}$$

Here $\mathcal{L}_{ce}$ represents the tutorial sequence cross-entropy loss; g represents generating tutorial semantic vector; Q_t represents the path adaptation quality score; Let δ denote the lowest quality threshold; Θ denotes the model parameters; λ_1 to λ_4 are the loss weights. The objective function integrates the generation quality and learning path adaptation into the same training framework, which provides a stable basis for subsequent feedback evaluation, and provides a stable calculation basis and interface for subsequent path selection feedback prediction and system deployment verification.

This model completes the modeling of digital media tasks from input to tutorial generation. Retrieval enhancement and constraint decoding jointly control the content of the tutorial, so that the generated results can correspond to the real operation steps and the process of work modification, and provide input for subsequent path adaptation and experience evaluation.

3.2 Knowledge constraint and retrieval enhancement mechanism for tutorial content generation

This module deals with knowledge boundary, operation executability and source traceability in the process of tutorial content generation. Digital media courses contain heterogeneous

knowledge such as layer editing, shot editing, interactive scripts, export specifications, etc. Relying solely on generative models is easy to cause step skipping, inconsistent tool parameters, or examples off task. The system integrates the course knowledge graph, tool command base, work sample base and error case base into the retrieval space. The evidence recall is completed before generation, the constraint verification is completed during generation, and the consistency score is completed after generation.

To illustrate the synergistic relationship between knowledge constraints and retrieval enhancement in tutorial generation, Fig. 2 shows the processing flow from the entry of task semantics into the knowledge base to the writeback of generated results. The left side shows task status and learner needs, the middle side shows two-channel retrieval, knowledge filtering, constraint decoding and evidence verification, and the right side outputs step-by-step tutorials, example resources and modification suggestions. The structure enables the tutorial content to correspond to real operational links of digital media projects.

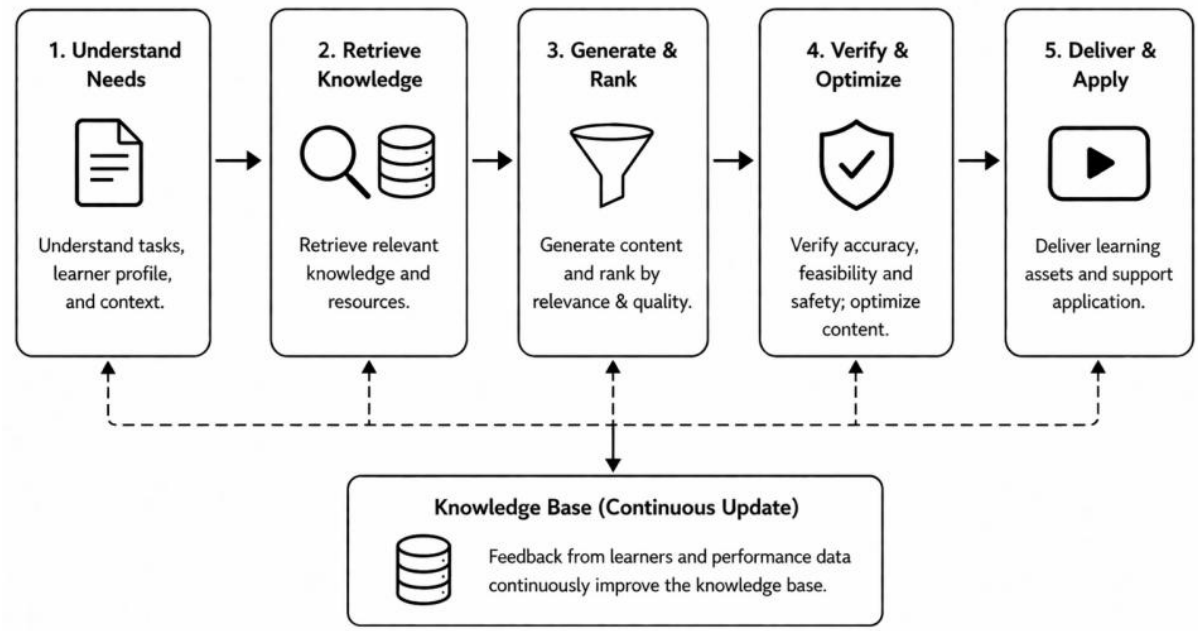

Figure 2: Tutorial content generation process driven by knowledge constraints and retrieval enhancement

In order to enable the course knowledge to be accurately invoked by the generation model, the system organizes the knowledge points, tool commands and work cases into a weighted graph structure, and uses the dependency relationship and semantic similarity to jointly determine the edge weight, which is calculated as follows:

$$A_{mn} = \sigma(\theta_1 d_{mn}^{dep} + \theta_2 \cos(v_m, v_n) + \theta_3 c_{mn}^{tool}) \tag{6}$$

where, A_{mn} represents the association weight between knowledge node m and node n. d_{mn}^{dep} represents the dependency strength of course prior knowledge. v_m and v_n represent node semantic vectors. c_{mn}^{tool} represents the co-occurrence strength of two nodes in the same software operation chain. The parameters θ_1 to θ_3 are trainable parameters. This formula makes the knowledge constraints not only come from chapter order, but also reflect the continuous operation relationship in digital media software.

In the evidence recall stage, the system uses sparse text matching and dense semantic retrieval at the same time, and adds matching items in the task stage to avoid the generation

model quoting information inconsistent with the current work status. The fusion retrieval probability is shown in the following equation:

$$P(r_j|z_t) = \frac{\exp(\gamma B_j + (1-\gamma) z_t^\top r_j + \mu \phi_j)}{\sum_{l=1}^{N} \exp(\gamma B_l + (1-\gamma) z_t^\top r_l + \mu \phi_l)} \tag{7}$$

Here, $P(r_j|z_t)$ represents the probability that fragment r_j is selected as evidence generation. B_j represents the BM25 text matching score; $z_t^\top$ represents the semantic similarity between the task state and the evidence vector; Let ϕ_j denote the matching value between the evidence and the current task phase; γ and μ are the weight coefficients. This formula can limit the search scope to interpretable and reviewable curriculum resources.

In order to prevent tool name errors, parameter ranges out of bounds, or step sequence breaks in the tutorial output, the system sets constraint check vectors for each generation step, and calculates the pass degree according to command legitimacy, parameter legitimacy, and order legitimacy, as shown in the following equation:

$$\kappa_i = \prod_{q=1}^{Q} \mathbb{I}(a_{i,q} \in \mathcal{C}_q) \cdot \exp(-\|s_i - s_{i-1} - \Delta_i\|_1) \tag{8}$$

Here, κ_i denotes the constraint passability of the i tutorial step. $a_{i,q}$ denotes the type q operation element in the step; $\mathcal{C}_q$ denotes the corresponding legal set; s_i and s_{i-1} denote the neighboring step states; Let Δ_i denote the expected state change. This formula combines discrete rules and continuous state deviations into the verification at the same time, which makes the tutorial steps closer to real software operations.

In the training phase, the retrieval evidence coverage, generation semantic consistency and constraint passing degree should be merged into the same goal, so that the model can generate fluent text while retaining evidence sources and task boundaries. The optimization goal is as follows:

$$\mathcal{J} = -\sum_{i=1}^{L} \log P(y_i|R_t) + \lambda_a(1-\bar{\kappa}) + \lambda_b \max(0, \xi - \mathrm{Cov}(R_t, Y)) \tag{9}$$

where $\mathcal{J}$ represents the knowledge enhancement generation target; $P(y_i|R_t)$ represents the probability of generating the i lemma based on the retrieved evidence. $\bar{\kappa}$ is the average constraint passability of all steps. $\mathrm{Cov}(R_t, Y)$ represents the coverage degree of the generated content on the retrieval evidence. Let ξ denote the lowest coverage threshold; λ_a,λ_b are the tuning parameters. This objective is able to suppress evidence-free generation and preserve the professional boundaries of tutorial content.

After the above processing, the knowledge constraint and retrieval enhancement mechanism transformed the course resources, operation specifications and work cases into callable evidence. The generative model is no longer responsible for knowledge judgment alone, but outputs the tutorial content under the combined effect of retrieval results and constraint verification, which provides stable input for subsequent learner state recognition and path adaptation. The mechanism also keeps the evidence number, call time and parameter source, which is convenient for teachers to review the tutorial generation process, and it is also convenient for the system to track the source of errors in new projects. In the face of different tasks of images, videos and interactive media, the model can automatically narrow the generation range according to the knowledge node and operation state, so as to improve the stability, interpretability and engineering deployment value of the tutorial content, and

support the continuous update of the library.

3.3 Learner state recognition and dynamic tutorial path adaptation algorithm

This module is used to convert the evidence of the learner's actions in the digital media project into a state judgment, and select the next tutorial path based on it. The system receives software log, work version difference, prompt input, tutorial adoption record and stage score, and forms the learner representation through the state recognition network, and then outputs the next step guidance in image design, video production or interactive media development by the path adaptation algorithm. This process does not rely on a fixed chapter order, but dynamically adjusts the tutorial difficulty according to the project progress and error types.

To illustrate the computational relationship between learner state recognition and path adaptation, Fig. 3 shows the running link from learning evidence input to tutorial path output. Multi-source learning evidence is shown on the left, state encoding, capability estimation, candidate path scoring, and strategy selection are shown in the middle, and the next step tutorial, supplementary examples, and fallback hints are output on the right. This structure keeps the generated tutorial in sync with the real authoring process.

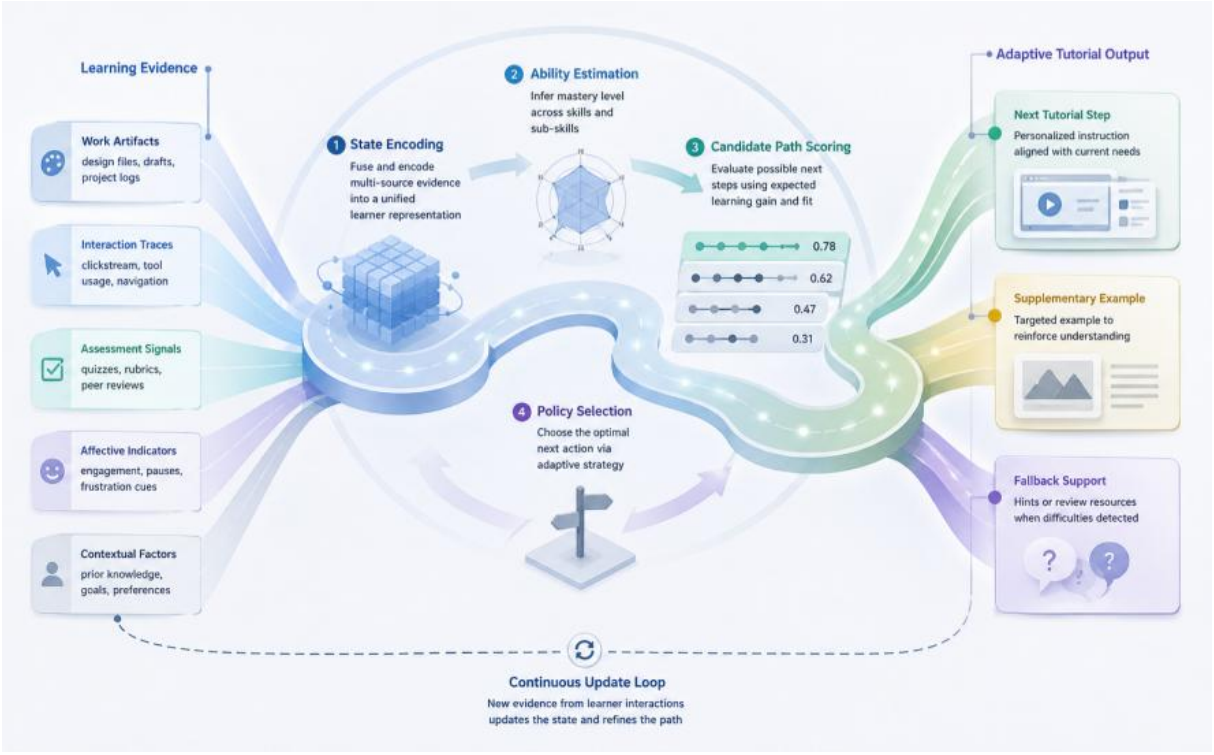

Figure 3: Learner state identification and dynamic tutorial path adaptation process

In order to avoid the deviation of state recognition caused by a single log, the system puts operation behavior, project difference, text intention and feedback results into the control fusion unit, and retains the contribution of different evidence to the state vector. The learning evidence aggregation is shown in the following equation:

$$m_t = \sigma(W_g[l_t; d_t; p_t; b_t]) \odot \tanh(W_s[l_t; d_t; p_t; b_t] + c) \quad (10)$$

where, m_t represents the learner state representation after the t round of interaction. l_t

stands for software operation log embedding. d_t stands for difference of works version; p_t represents student prompt semantics; b_t is feedback adoption behavior; W_g,W_s are mapping matrices. This formula filters the weak correlation evidence through the gating term, so that the state recognition is more suitable for the work modification process.

In order to map the learner state to the interpretable course ability node, the model matches the state representation with the digital media knowledge node and generates the mastery probability of different skill dimensions. The ability distribution is calculated as follows:

$$M_t = \mathrm{Softmax}\left(\frac{K^\top m_t + b_k}{\sqrt{d_k}}\right) \tag{11}$$

where, M_t represents the probability distribution of learners at each ability node. K represents the course knowledge node matrix; b_k represents the node bias; d_k denotes the node vector dimension. The formula can identify students' weak positions in color adjustment, lens cohesion, interaction logic and derived specifications, and provide an interpretable basis for path adaptation.

In order to select the next tutorial that matches the current state, the system simultaneously considers the ability gap, the benefit of the work, the operation cost and the repeated learning penalty, so that the recommendation results can enter the actual creation process. The candidate path score is shown in the following equation:

$$V(a_t) = \omega_1 G(a_t, M_t) + \omega_2 \Delta Q(a_t) - \omega_3 C(a_t) - \omega_4 R(a_t, h_t) \tag{12}$$

where $V(a_t)$ represents the comprehensive score of the a_t of the candidate tutorial path; $G(a_t, M_t)$ represents the matching value between the path and the capacity gap. $\Delta Q(a_t)$ represents the expected quality gain of the work; $C(a_t)$ is the cost of operation.$R(a_t, h_t)$ is the history duplication penalty. ω_1 to ω_4 are the weights. This formula makes the path selection take into account both learning needs and operational burden.

In order to maintain the stability of path recommendation, the expert path, work quality prediction error and strategy fluctuation are simultaneously constrained in the training phase, so that the model can output continuous and reliable tutorial paths in different digital media tasks. The objective function is shown in the following equation:

$$\mathcal{L}_{\mathrm{path}} = -\sum_{t=1}^{T} \log \pi(a_t^* | m_t) + \lambda \sum_{t=1}^{T} (\hat{q}_t - q_t)^2 + \beta D_{KL}(\pi_t || \pi_{t-1}) \tag{13}$$

Here, $\mathcal{L}_{\mathrm{path}}$ represents the path adaptation loss; a_t^* denotes the annotated or teacher-confirmed target path; Let $\pi(a_t^*|m_t)$ denote the probability that the model chooses this path. $\hat{q}_t$ and q_t denote the predicted and real work quality, respectively; D_{KL} is used to constrain neighboring policy differences. This objective function keeps the system coherent when dynamically adjusting the tutorial, and does not switch paths frequently due to short-term operation fluctuations.

The algorithm of learner state recognition and dynamic tutorial path adaptation completes the computational transformation from behavior evidence to tutorial selection. The operation log, work difference, prompt semantics and feedback adoption results were integrated as the learning state representation, and the ability node identification results further limited the scope of tutorial recommendation. The path adaptation module selects the next tutorial according to the change benefit of the work, the learning burden and the historical repetition,

so that the generated content can be adjusted to follow the actual advancement of the digital media project. The algorithm provides a continuous and traceable status basis for subsequent human-computer interaction feedback and learning experience evaluation.

3.4 Human-computer interaction feedback and evaluation mechanism of digital media learning experience

This module converts the interaction records between students and the generative tutorial system into learning experience evaluation results, and writes them back to the tutorial generation and path adaptation module. Learning has an iterative character of creation, reading tutorials, taking steps, undoing actions, asking questions again, and submitting work to form computable evidence. Man-machine feedback, work benefit, tutorial adoption and response delay are incorporated into the evaluation link, so that the evaluation results correspond to real projects.

The system first collates the interaction feedback field, as shown in Table 2. The fields in the table cover the behavioral course of the student from receiving the tutorial to performing the modification and provide input for experience scoring and policy updates.

Table 2: HCI feedback fields and computational purposes

Field	Recorded Content	Purpose
Dwell Time	Reading time for tutorial steps	Readability judgment
Adoption Marker	Whether the operation is completed	Effectiveness judgment
Rollback Count	Number of undo or return actions	Cognitive load judgment
Query Text	Follow-up questions from learners	Deviation judgment
Version Difference	Changes in the submitted work	Benefit judgment

In order to depict the learners' immediate response to the generated tutorial, the system converts the stay, adopt, backtrack, inquiry and work difference into the feedback intensity. The feedback aggregation process is shown in the following equation:

$$\eta_t = \sigma(w^\top[\ell_t, a_t, r_t, g_t, \Delta v_t] + b) \tag{14}$$

Here, η_t represents the feedback strength of the t interaction. ℓ_t denotes the stay feature; a_t represents the value of acceptance; r_t is the back-off frequency. g_t represents question semantics; Δv_t is version difference; Let σ denote the Sigmoid function. This formulation compreszes discrete behavior and continuous work changes into the same feedback scale, which facilitates the judgment of whether the tutorial enters the operation.

In the experience evaluation stage, the system considers the tutorial relevance, operation benefit, response speed and cognitive burden, and the comprehensive experience score is shown in the following equation:

$$E_t = \sum_{i=1}^{4} \omega_i \tilde{x}_{t,i} - \omega_5 \log(1 + \tau_t) - \omega_6 h_t \tag{15}$$

where E_t represents the experience score; $\tilde{x}_{t,i}$ denotes the normalized relevance, adoption rate, work benefit, and feedback clarity; τ_t is the response delay. h_t stands for repeated burden; Let ω_i be the weight coefficients. This formula avoids measuring experience only by subjective evaluation, and makes the evaluation result close to the operation cost of digital media projects.

The evaluation index structure is shown in Table 3. This table puts the interaction behavior, generation quality and project results in the same framework, which is easy to compare in the experimental stage.

Table 3: Evaluation metrics for digital media learning experience

Metric	Data Source	Meaning
Relevance	Matching between task and tutorial	Content alignment
Path Matching Rate	Recommended path and annotated path	Adaptation accuracy
Work Benefit Rate	Score changes before and after revision	Guidance value
Feedback Error	System score and human score	Evaluation reliability
Response Latency	System logs	Deployment efficiency

In order to verify the consistency between the generated tutorial and the actual operation results, the system takes the tutorial semantic vector, execution log and work revenue into the verification at the same time, and the feedback consistency calculation is shown in the following equation：

$$C_t = \frac{\cos(y_t, o_t) + \cos(y_t, d_t)}{2} \cdot \exp(-|\hat{b}_t - b_t|) \tag{16}$$

where C_t represents feedback consistency; y_t represents generating tutorial semantic vector; o_t stands for execution log embedding. d_t stands for works differential embedding; $\hat{b}_t$ and b_t denote the system predicted score versus the human score. This formulation configures language content, operation execution, and work outcome, and is able to identify tutorial outputs that are ostensibly relevant but have insufficient revenue.

In order to make the feedback result enter the next round of path adaptation, the system updates the tutorial strategy weight according to the experience score and consistency result. The strategy update process is as follows:

$$\pi_{t+1}(a) = \frac{\pi_t(a)\exp(\lambda E_t C_t - \zeta\Omega(a))}{\sum_{a' \in \mathcal{A}} \pi_t(a')\exp(\lambda E_t C_t - \zeta\Omega(a'))} \tag{17}$$

where $\pi_{t+1}(a)$ denotes the probability of selecting the tutorial action a in the next round; E_t is the experience score. C_t represents feedback consistency; $\Omega(a)$ is the action complexity penalty. λ and ζ are the tuning parameters. This formula preserves the effective path and suppresses the high-burden tutorial, so that the subsequent generated content is appropriate to the learner's state.

The mechanism completes the closed-loop processing after generation. The system judges the tutorial quality from four types of evidence: reading, executing, modifying and evaluating, and writes the results back to the path adaptation and generation control module. Images, videos, and interactive items thus receive continuous feedback and provide quantitative metrics for the experimental section.

4 Analysis of results

4.1 Implementation Environment and experimental setup

This experiment builds the DGAI-Tutor test environment for university digital media courses. The data comes from the learning records of 128 students for 12 consecutive weeks, covering

three task types: image design, video production, and interactive media, resulting in 6420 labeled learning segments. Each segment consisted of software operation logs, work version differencing, prompt input, tutorial adoption records, peer feedback, and instructor ratings. Anonymization processing, timestamp alignment, abnormal log removal and project version hash mapping were completed before data entered the model, and then the training set, validation set and test set were divided by 7:2:1. The text prompt is encoded by Sentence-BERT, the metadata of works is encoded by visual feature extractor and version difference operator, and the operation sequence is modeled by Transformer temporal encoder. The generation side uses retrieval enhanced large language model, and the knowledge base includes course knowledge graph, tool command library, excellent work examples and common error fragments. The experimental environment is shown in Table 4.

Table 4: DGAI-Tutor experimental environment and data setup

Item	Setting
Data Scale	128 students and 6,420 learning segments
Task Types	Image design, video production, and interactive media
Data Sources	Operation logs, version differences, prompt texts, and feedback records
Model Modules	RAG, Transformer encoder, and path adapter
Software Environment	Python 3.10, PyTorch 2.1, FAISS, and PostgreSQL
Hardware Environment	RTX 4090 GPU, 64 GB RAM, and Intel i9

The experiment uses a unified data partition method and random seed to compare three types of methods: static tutorial recommendation, rule retrieval generation and DGAI-Tutor. The model training batch is set to 32, the maximum input length is set to 1024 lemmas, the learning rate is set to 2e-5, and AdamW is used as the optimizer. The evaluation metrics include tutorial relevance, path matching rate, feedback MAE, and average response delay. In order to ensure that the results can be reviewed, the system saves the retrieval evidence, constraint check record and student acceptance status generated each time, and retains the five-fold cross-validation results. Indicators were calculated separately for different task types to avoid a single work type affecting the overall judgment. The image task records layers, color schemes, and export specifications, the video task records shots, audio tracks, and transitions, and the interactive media task records page nodes, script calls, and event responses. The above records are saved synchronously with the human scoring to examine the stability and interpretability of the model in different authoring stages. No additional amplified samples were introduced in the testing phase, and the experimental process was consistent.

4.2 Model performance evaluation and comparative analysis

The performance evaluation of the model focuses on "whether the tutorial matches the task, whether the path conforms to the creation process, whether the feedback is close to human judgment, and whether the system has the ability of real-time response" in digital media learning. The test set contains three categories of tasks: image design, video production, and interactive media. The comparison methods include static tutorial recommendation, rule retrieval generation, and DGAI-Tutor. Static tutorial recommendation mainly pushes fixed resources based on course chapters, and rule retrieval generation relies on keyword matching and template step combination. DGAI-Tutor introduces retrieval enhancement, knowledge constraint, state recognition and feedback writeback mechanisms. The evaluation indicators

include tutorial relevance, path matching rate, feedback MAE and average response delay, which can reflect the generation quality, path adaptation effect, evaluation error and system deployment efficiency at the same time.

To observe the overall differences of the three types of methods in multiple indicators, Fig. 4 presents the comprehensive performance after normalization using radar charts. DGAI-Tutor achieves 92.8, 89.6, 91.4 and 88.7 in tutorial relevance, path matching rate, feedback error reverse score and delay reverse score, respectively. Rule retrieval generation corresponds to 84.1, 78.3, 82.6 and 81.5; Static tutorial recommendations correspond to 76.4, 69.8, 74.2, and 90.1. The static method has fast response speed, but the content is difficult to follow the changes of the work. Rule retrieval generation can return some relevant information, but the understanding of the continuous creation process is insufficient. DGAI-Tutor maintains a relatively balanced performance in generation quality and path adaptation.

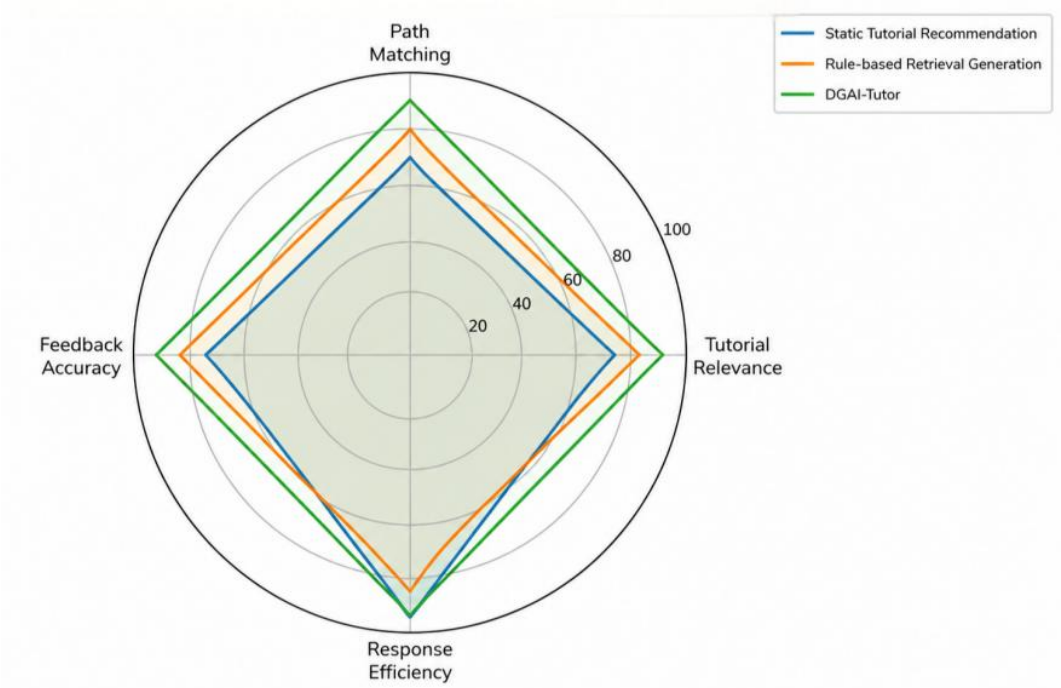

Figure 4: Radar chart of comprehensive performance of three classes of tutorial models

The manual review results were used to verify the reliability of the scoring labels. The three teachers scored from the three dimensions of task fit, step executability and work modification value respectively, and Table 5 shows the consistency of review under different task types. The consistency of the image design task is the highest, the video production has some differences in the shot rhythm and the judgment of sound and painting synchronization, and the interactive media task has a slightly higher score difference due to the more complex script logic, page nodes and event trigger relationships. The overall Cohen's κ reaches 0.82, which indicates that the artificial labels have good stability and can be used for model performance evaluation.

Table 5: Manual review consistency statistics

Task Type	Sample Size	Average Scoring Difference	Cohen's κ	Review Conclusion
Image Design	2,180	0.18	0.86	High consistency
Video Production	2,070	0.24	0.81	Differences exist in rhythm judgment
Interactive Media	2,170	0.27	0.79	Slightly higher divergence in logic scoring
Overall	6,420	0.23	0.82	Suitable for model evaluation

Error type analysis is used to determine the source of failure of generated tutorials. Fig. 5 adopts heat maps to present the major error proportions in the three categories of tasks. The error of parameter range in image design is 12.6%, the error of step sequence in video production is 15.4%, and the omission of event logic in interactive media is 17.8%. The proportion of invalid tutorials in DGAI-Tutor is all lower than 8%, indicating that retrieving evidence and constraint-based decoding are able to reduce unsupported instructions, step jumps, and parameter out-of-bounds. The errors for interactive media tasks are concentrated on event binding and page skipping, indicating that more fine-grained semantic checks of scripts are still needed for complex projects.

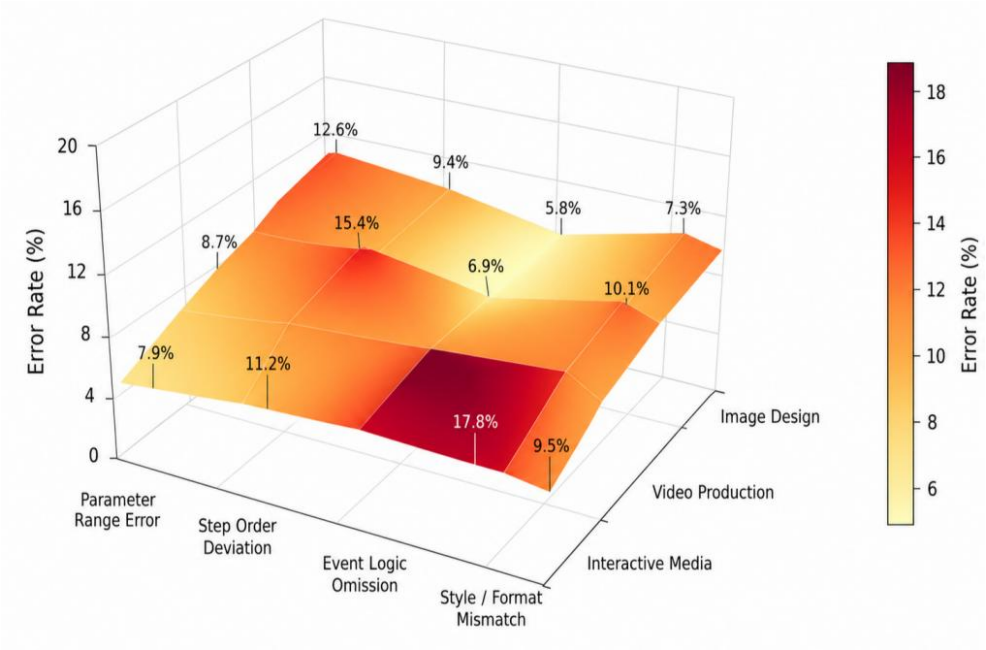

Figure 5: Heatmaps of tutorial error types for different digital media tasks

Response efficiency determines whether the tutorial system can be embedded in the real-time authoring process of students. Fig. 6 uses box plots to show the response delay distribution of the three categories of tasks. The median latency was 68ms for the image design task, 76ms for video production, and 81ms for interactive media, with 95th quantile latency below 126ms for all three categories. The latency differences mainly come from the number of retrieved fragments, the number of constraint validation times, and the syntax checking overhead of code-class tutorials. The overall results show that DGAI-Tutor is able to maintain continuous interactions on the classroom side without significantly interrupting student project operations.

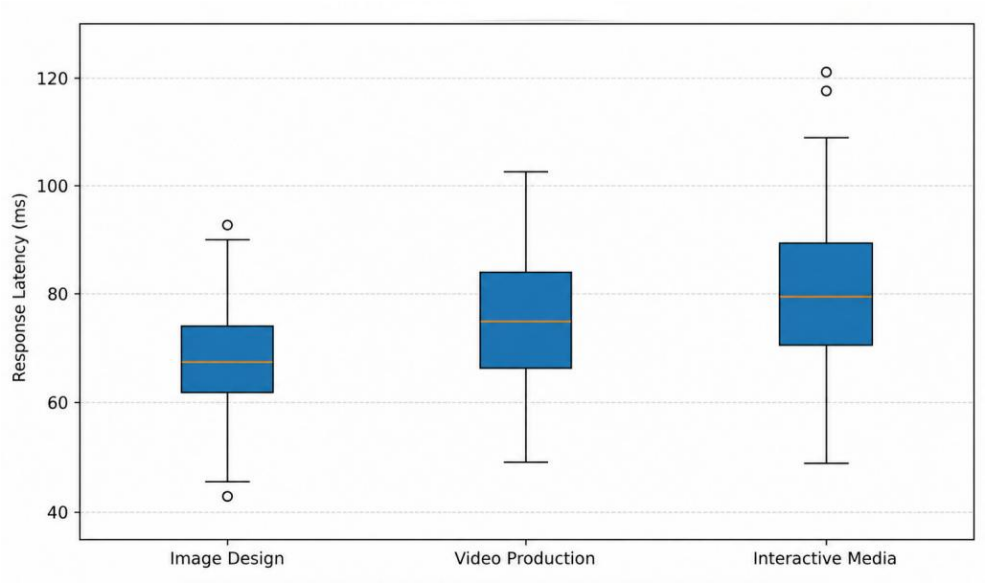

Figure 6: Boxplots of response latency for different task types

To verify the stability of the performance differences, Table 6 presents the statistical test results of DGAI-Tutor and the two types of baseline methods on the core indicators. Tutorial relevance, path matching rate, and feedback MAE were tested using paired t-tests with p values below 0.01. The results show that the advantage of DGAI-Tutor over static tutorial recommendation and rule retrieval generation does not come from a single sample division, but is related to state recognition, retrieval enhancement, and feedback writeback in the model structure.

Table 6: Significance tests between DGAI-Tutor and baseline methods

Comparison Object	Metric	Test Method	Statistic	p-value	Result
DGAI-Tutor vs. Static Tutorial Recommendation	Tutorial Relevance	Paired t-test	6.42	0.0002	Significant
DGAI-Tutor vs. Static Tutorial Recommendation	Path Matching Rate	Paired t-test	7.15	0.0001	Significant
DGAI-Tutor vs. Static Tutorial Recommendation	Feedback MAE	Paired t-test	5.87	0.0004	Significant
DGAI-Tutor vs. Rule-based Retrieval Generation	Tutorial Relevance	Paired t-test	4.31	0.0015	Significant
DGAI-Tutor vs. Rule-based Retrieval Generation	Path Matching Rate	Paired t-test	4.96	0.0008	Significant
DGAI-Tutor vs. Rule-based Retrieval Generation	Feedback MAE	Paired t-test	3.74	0.0032	Significant

The training process is shown in Fig. 7. In DGAI-Tutor, the validation loss decreased slowly after the 12th round and entered the stable interval after the 18th round, resulting in a final validation loss of 0.184. The static tutorial recommendation model has limited gains in path matching after the 10th round, and rule retrieval generation mainly depends on threshold adjustment, with more obvious fluctuations. This result shows that it is difficult to deal with continuous changes in digital media projects by only relying on fixed resources or keyword recall, while state encoding and feedback write-back can make the model gradually form a

stable tutorial generation boundary.

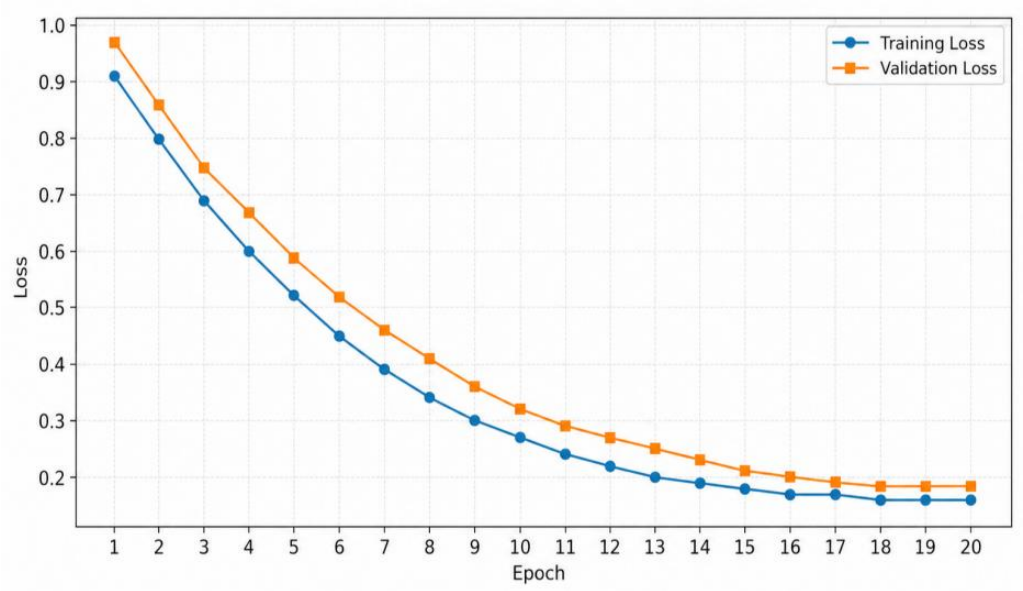

Figure 7: Model training versus validation loss change curve

The learning experience prediction error is shown in Fig. 8. The feedback MAE of DGAI-Tutor in image design, video production and interactive media tasks is 0.196, 0.221 and 0.235, respectively, and the overall value is 0.214. The overall MAE for rule retrieval generation is 0.287 and 0.342 for static tutorial recommendation. The DGAI-Tutor error is lower, which indicates that the system feedback is closer to the human rating. This result is consistent with the path matching rate in the previous section, indicating that the feedback module can judge the tutorial effect based on the work changes and interaction records.

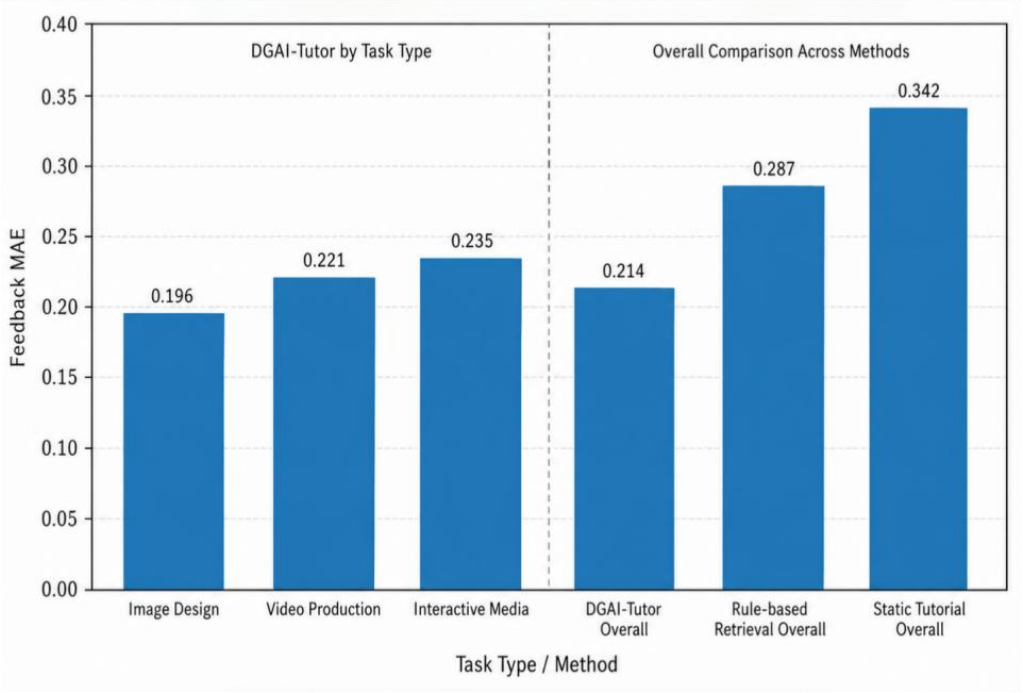

Figure 8: Feedback MAE distribution plots for different task types

Ablation experiments are used to judge the contribution of each module to the overall performance, and Table 7 shows the results after removing different components. After removing the knowledge constraints, the tutorial correlation decreased to 87.2%. After removing the path adapter, the path matching rate decreases to 80.4%. After removing the feedback writeback, the feedback MAE increases to 0.291. After removing retrieval enhancement, the proportion of incorrect tutorials reaches 13.6%. The complete DGAI-Tutor maintains 92.8% tutorial relevance, 89.6% path matching rate, and 0.214 feedback MAE, indicating that retrieval, constraint, state recognition, and feedback write-back jointly support model performance.

Table 7: DGAI-Tutor ablation experiment results

Model Setting	Tutorial Relevance / %	Path Matching Rate / %	Feedback MAE ↓	Incorrect Tutorial Ratio / % ↓
Without Knowledge Constraints	87.2	86.1	0.247	10.8
Without Path Adapter	90.4	80.4	0.238	9.6
Without Feedback Write-back	91.1	84.7	0.291	8.9
Without Retrieval Augmentation	85.6	82.5	0.263	13.6
Complete DGAI-Tutor	92.8	89.6	0.214	6.7

The comprehensive results show that DGAI-Tutor has better generation quality, path adaptation ability and feedback stability in digital media learning scenarios. The model can convert learners' actions, work versions, prompt semantics and course knowledge into computable evidence, and form a closed loop between tutorial generation, recommendation ranking and experience evaluation. Compared with static tutorial recommendation and rule retrieval generation, DGAI-Tutor not only improves the metric performance, but also enables the generated content to correspond to real authoring tasks. The results in the three categories of tasks, image design, video production and interactive media, together illustrate that the model is suitable for dynamic tutorial support in open digital media projects.

5 Discussion

The experimental results of DGAI-Tutor show that the dynamic generative tutorial model has better task fitting ability in digital media learning scenarios. Compared with the static tutorial recommendation, the model does not push resources according to the chapter directory, but converts the prompt text, work version, operation log and feedback acceptance record into computable states, and then the generation boundary is limited by retrieval enhancement and knowledge constraints. The high correlation in the image design task indicates that the model is able to recognize the operational relationships between layers, color schemes, and derived specifications. The path matching results in the video production task show that the timeline, transition, audio track and shot rhythm can be incorporated into the sequence modeling. Although the interactive media task has a slightly higher feedback error, it still generates executable suggestions based on page nodes, script calls, and event responses. Ablation experiments show that it is difficult to rely on retrieval or fixed rules alone to support authoring, and a closed loop needs to be formed between knowledge constraints, state

recognition and feedback writeback. This result shows that the value of generative AI in digital media teaching does not lie in replacing teacher explanation, but in organizing scattered creation evidence into continuous tutorials, so that the revision, back-off and re-questioning in the learning process can enter the next round of generative judgment. The performance of the model on response delay shows that retrieval, generation, and verification can form a usable link. Instead of generalizing answers, students were given hints related to file states, task phases, and operational limitations at the creation site. After keeping the generation evidence and the adoption record, the teacher can review the system output and determine whether the student is stuck in the tool use, creative expression or project organization.

6 Conclusions

This paper constructs a dynamic generative tutorial model of DGAI-Tutor for university digital media learning, which forms a complete computing link around task semantic encoding, knowledge constraint retrieval, learner state recognition and interactive feedback evaluation. The model can transform the operational evidence in image design, video production and interactive media projects into the basis for tutorial generation, and maintain the continuity of the creation process in the path adaptation. The research still has some limitations. The experimental samples are mainly from internal projects of digital media courses. Although the task types cover common creation scenarios, the generalization ability of cross-school, cross-software platform and cross-professional courses still needs to be verified. The score of work quality depends on teachers' review, and the score standard may still be affected by style preference in open creation. The retrieval enhancement module depends on the completeness of the course knowledge base, and when the tool version is updated or the project requirements change, the knowledge nodes need to be maintained synchronously. Future research can introduce cross-platform plug-in acquisition interface to expand 3D animation, game interaction and virtual reality task data, and combine lightweight large model, online incremental learning and federal privacy protection mechanism to enhance the stability and scalability of the system in real teaching deployment. In the future, the evidence labeling method of generated content should be refined, and each step of the tutorial should be mapped to knowledge fragments, software commands, and work versions. At the system deployment level, edge caching and local reasoning modules can be added to reduce the impact of network fluctuations on classroom interaction. The evaluation level can add student self-evaluation, peer review and long-term work archives, so that the experience judgment from a single feedback to a continuous growth record, and reduce maintenance costs at the same time.

Funding

This work was supported by the Hubei Engineering University Teaching Reform Research Project (Grant No. 2024JY002).

References

[1] Alier M, Peñalvo F J G, Camba J D. Generative Artificial Intelligence in Education: From Deceptive to Disruptive[J]. International Journal of interactive multimedia and artificial intelligence, 2024, 8(5): 5-14.

[2] Peñalvo F J G, Ingelmo A V. What do we mean by GenAI? A systematic mapping of the evolution, trends, and techniques involved in Generative AI[J]. IJIMAI, 2023, 8(4): 7-16.

[3] Kasneci E, Seßler K, Küchemann S, et al. ChatGPT for good? On opportunities and challenges of large language models for education[J]. Learning and individual differences, 2023, 103: 102274.

[4] Ray P P. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope[J]. Internet of things and cyber-physical systems, 2023, 3: 121-154.

[5] Guettala M, Bourekkache S, Kazar O, et al. Generative artificial intelligence in education: Advancing adaptive and personalized learning[J]. Acta Informatica Pragensia, 2024, 13(3): 460-489.

[6] Minn S. AI-assisted knowledge assessment techniques for adaptive learning environments[J]. Computers and Education: Artificial Intelligence, 2022, 3: 100050.

[7] Southworth J, Migliaccio K, Glover J, et al. Developing a model for AI Across the curriculum: Transforming the higher education landscape via innovation in AI literacy[J]. Computers and Education: Artificial Intelligence, 2023, 4: 100127.

[8] Verma N, Getenet S, Dann C, et al. Designing an artificial intelligence tool to understand student engagement based on teacher's behaviours and movements in video conferencing[J]. Computers and Education: Artificial Intelligence, 2023, 5: 100187.

[9] Hossen M K, Uddin M S. Attention monitoring of students during online classes using XGBoost classifier[J]. Computers and Education: Artificial Intelligence, 2023, 5: 100191.

[10] Dakakni D, Safa N. Artificial intelligence in the L2 classroom: Implications and challenges on ethics and equity in higher education: A 21st century Pandora's box[J]. Computers and Education: Artificial Intelligence, 2023, 5: 100179.

[11] Segbenya M, Bervell B, Frimpong-Manso E, et al. Artificial intelligence in higher education: Modelling the antecedents of artificial intelligence usage and effects on 21st century employability skills among postgraduate students in Ghana[J]. Computers and Education: Artificial Intelligence, 2023, 5: 100188.

[12] Habibi A, Muhaimin M, Danibao B K, et al. ChatGPT in higher education learning: Acceptance and use[J]. Computers and Education: Artificial Intelligence, 2023, 5: 100190.

[13] Albayati H. Investigating undergraduate students' perceptions and awareness of using ChatGPT as a regular assistance tool: A user acceptance perspective study[J]. Computers and Education: Artificial Intelligence, 2024, 6: 100203.

[14] Einarsson H, Lund S H, Jónsdóttir A H. Application of ChatGPT for automated problem reframing across academic domains[J]. Computers and Education: Artificial Intelligence, 2024, 6: 100194.

[15] Al Shloul T, Mazhar T, Abbas Q, et al. Role of activity-based learning and ChatGPT on students' performance in education[J]. Computers and Education: Artificial Intelligence, 2024, 6: 100219.

[16] Alnasyan B, Basheri M, Alassafi M. The power of Deep Learning techniques for predicting student performance in Virtual Learning Environments: A systematic literature review[J]. Computers and Education: Artificial Intelligence, 2024, 6: 100231.

[17] Baek C, Tate T, Warschauer M. “ChatGPT seems too good to be true”: College students’ use and perceptions of generative AI[J]. Computers and Education: Artificial Intelligence, 2024, 7: 100294.

[18] Baig M I, Yadegaridehkordi E. ChatGPT in the higher education: A systematic literature review and research challenges[J]. International journal of educational research, 2024, 127: 102411.

[19] Farrokhnia M, Banihashem S K, Noroozi O, et al. A SWOT analysis of ChatGPT: Implications for educational practice and research[J]. Innovations in education and teaching international, 2024, 61(3): 460-474.

[20] Cotton D R E, Cotton P A, Shipway J R. Chatting and cheating: Ensuring academic integrity in the era of ChatGPT[J]. Innovations in education and teaching international, 2024, 61(2): 228-239.

[21] Kamalov F, Santandreu Calonge D, Gurrib I. New era of artificial intelligence in education: Towards a sustainable multifaceted revolution[J]. Sustainability, 2023, 15(16): 12451.