ITM Web of Conferences **79**, 01035 (2025) https://doi.org/10.1051/itmconf/20257901035
KEIS-2025

Sim-GAIL with SAINT for Knowledge Completion in Intelligent Tutoring Systems

Mohammed hsan[1], *Pooja* Nayak. S[2], *Keerthi* Kumar. M[3*], *Loovesh* Sharma Ramwodin[4], and *Sarala.* D V[5]

[1]Department of Computers Techniques Engineering, College of Technical Engineering, The Islamic University, Najaf, Iraq
[2]Department of Information Science and Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, India
[3]Department of Electronics and Communication Engineering, Nitte Meenakshi Institute of Technology, Nitte (Deemed to be University), Bengaluru, India
[4]Department of Business Informatics and Software Engineering, University of Technology, Mauritius
[5]Department of Computer Science and Engineering, BMS College of Engineering, Bengaluru, India

Abstract. In recent times, learner profiles and knowledge graphs in educational platforms have gaps due to missing interactions or unseen concepts; knowledge completion helps fill missing parts and understand learners. Conventional evaluation methods depend poorly on targeted questions, which is insignificant in uncovering a student's true understanding, even though they demand long hours of testing. The continuous integration of Artificial Intelligence (AI) into digital education involves an advanced Intelligent Tutoring Systems (ITS), online course delivery, and learning management platforms. Traditional simulation methods fail to produce high-quality and diverse data, limiting their usefulness in training Intelligent Tutoring Systems (ITS). While ITS initiates learning paths, it faces challenges, such as cold-start issues, time demands, manual effort, and high costs. The proposed Sim-GAIL (Generative Adversarial Imitation Learning) integrates SAINT (Self-Attentive Neural Knowledge Tracing) to combine simulated student trajectory generation with attentive control prediction. The preprocessing step extracts raw interactions into trajectory elements based on actions and temporal patterns, enabling policy learning within a Markov Decision Process (MDP). Sim-GAIL generates realistic learning sequences to address cold-start limitations, whereas SAINT uses self-attention to trace evolving knowledge states for an accurate performance prediction. Experimental results show that the integrated Sim-GAIL with SAINT method performs better than the existing Item Response Theory (IRT) with the XGBOOST model for both the EdNet and ASSISTment17 datasets, achieving 98.42% and 98.71 for the Area Under the Curve (AUC) metric.

1 Introduction

The past decade has witnessed tremendous research in the field of Adaptive E-Learning Systems [1]. The use of artificial intelligence (AI) in education is not a new academic topic, as its use has been studied over the past 30 years [2]. With the increased significance of AI in learning, this study examines views, attitudes and behavioral intentions of pre-service instructors with reference to the implementation of AI based applications [3] [4]. The Intelligent Tutoring Systems (ITSs) are systems offering adaptive learning opportunities to students and normally capture process information as students interact with the learning content offered by the system [5]. Conventionally, ITSs are based on rule algorithms and pre-existing content. These systems may work well in a structured setting, but they lack the adaptability required to suit a variety of educational settings and the dynamic needs of learners [6].

In recent decades, rapid technological advancements have led to the proliferation of digitally enabled tools and services. Societal progress depends on the acquisition and dissemination of valuable knowledge [7]. However, the absence of the necessary infrastructure to support the process of data collection to support educational interactions prevents the implementation and value of AIED applications in most areas [8]. Individual learning is a practical way of enhancing the performance of learners. This strategy enables students to acquire learning items and strategies that meet their interests, needs, and learning styles. One-size-fits-all paradigm in e-learning is absurd since it disregards the variation in the capacity, competencies, and traits of the students. Even though the idea of a personalized learning path has its potential, it presents a challenge to a teacher, who must dedicate much time and effort to decorate the content to fit individual students [9]. Thus, one of the possible solutions is artificial intelligence that would assist teachers in developing individual learning journeys. Educational data mining studies have previously indicated the use of different machine learning (ML) and deep learning (DL) models to predict student performance; nevertheless, the current methodologies are still characterized by dire limitations. Conventional ML models such as Logistic Regression and Support Vector Machines, presume linear associations between the features, which cannot

* Corresponding author: keerthi.kumar@nmit.ac.in

adequately represent the multidimensional and complex relationships among features that characterize student learning behaviors [4].

Recent studies have attempted to address these limitations. Cao, W et al [10] proposed a symmetric dual-network architecture with concept embedding and question selection networks using variational inference and symmetric reward structures. Hakkal S. and Ait Lahcen A [11], enhanced logistic regression-based models such as IRT and DAS3H were developed using XGBoost across eight ITS datasets. Zhu, M et al., [12] introduced MPSG, a Meta-Path Structured Graph method that captures high-order cross-entity relationships via intimacy-based sampling and self-supervised learning. Wang, D et al. [13] presented MEGR, a multibehavior-enhanced group recommendation model using specialized interaction subgraphs. Zhang, W et al.,[14] proposed GAKT-BF, a graph attention KT model with behavioral features, and question-question correlation graphs.

To overcome the remaining challenges in personalization, trajectory realism, and cold-start resolution, this research proposes a Sim-GAIL with SAINT framework. The contributions of the proposed Sim-GAIL to the SAINT framework are as follows:

- This Sim-GAIL with SAINT framework introduces a transformer-based module that encodes historical student behavior into latent states, enabling realistic and diverse trajectory generation for cold-start resolution.
- By combining generative adversarial imitation learning with TP-informed initialization, the model simulates expert-like student interactions without handcrafted reward functions, thereby improving fidelity and generalization.
- The system employs a dual-network architecture with symmetric concept embeddings and uncertainty-aware objectives, enhancing the diagnostic equivalence and performance in downstream KT tasks.

The remainder of the paper is organized as follows: Section 2 provides a detailed explanation of the proposed methodology, followed by results and discussion in Section 3. The conclusion is presented in Section 4

2 Proposed methodology

This section presents an integrated methodology combining Sim-GAIL, a generative adversarial imitation learning framework, with a trajectory prediction (TP) using SAINT module to simulate realistic student learning behavior in Intelligent Tutoring Systems (ITS). The proposed approach addresses key limitations in ITS training, including the cold-start problem, data sparsity, and need for high-fidelity synthetic trajectories. The integration leverages the strengths of both modules: Sim-GAIL's adversarial policy learning and TP's sequential prediction capabilities. The methodology is organized into six stages: data preprocessing, trajectory segmentation, feature construction, TP module design, Sim-GAIL with SAINT architecture, and integration strategy as shown in the Figure 1.

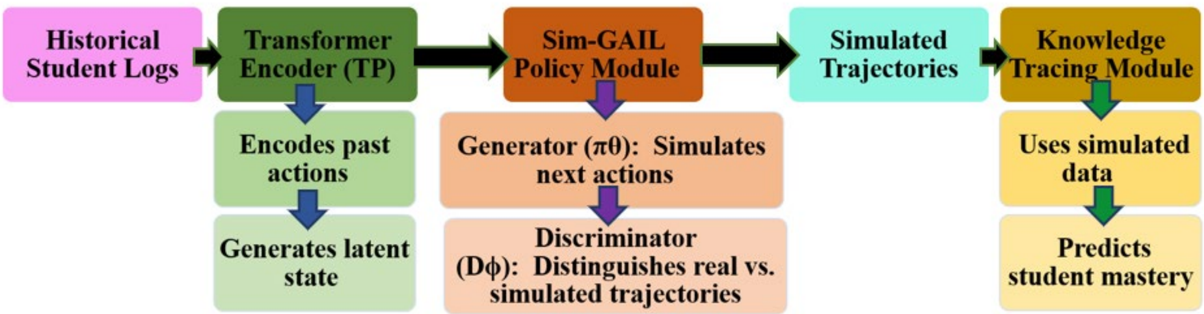

Fig. 1. The architecture of the proposed Sim-GAIL with SAINT framework.

2.1 Preprocessing for MDP-Based student simulation

The preprocessing step converts raw student interaction logs into structured trajectory data that can be used for policy learning within a Markov Decision Process (MDP). This involves identifying meaningful actions, constructing state representations, and organizing data into sequences that reflect real learning behavior over time.

2.1.1 Dataset overview

This research utilizes two prominent educational datasets: EdNet and ASSISTment17.

EdNet [15] served as the primary source of student interaction data. Collected from the SANTA platform, a TOEIC preparation system deployed in South Korea comprises over 131 million interaction logs from 784,309 students across 13,169 exercises. The dataset is hierarchically structured into four levels (KT1–KT4), each offering an increasing granularity of student actions. For this research, the KT2 and KT3 layers were employed, as they provide detailed records of question attempts, lecture views, and response behaviors. Additionally, the KT1 subset, containing 13.5 million interactions, was used for computational feasibility while preserving diversity across listening, reading, and grammar skills.

ASSISTments include student interaction data from an online tutoring system that focuses on mathematics. The 2012–2013 academic year subset contained

346,860 interactions from 5,549 students across 124 skills, following standard preprocessing protocols. Specifically, the ASSISTment17 dataset [16], also from the same platform, comprised 942,816 answer records from 1,709 students, covering 3,162 questions and 102 skills. Together, these datasets offer a rich and diverse foundation for modeling student learning behaviors and evaluating the performance of adaptive learning systems.

2.1.2 Action space construction

To define the action space, the preprocessing step groups the questions and lectures into conceptual domains. Following this method, the content is divided into seven groups based on the expert-defined domains. Within each group, the questions are further categorized into four difficulty levels based on the correctness rates. Lectures are assigned a default difficulty level of zero. This resulted in 35 distinct action types: four question levels, one lecture per group, and seven groups.

Each action in the trajectory is represented by an integer ranging from 1 to 35, corresponding to a specific question or lecture type. For example, actions 1–4 in Group 1 represent questions of increasing difficulty, whereas Action 5 represents the lecture. This structure ensures that the action space is comprehensive and consistent among students.

2.1.3 Temporal segmentation of trajectories

Student logs are segmented into learning trajectories based on temporal gaps and activity types. A trajectory is defined as the sequence of actions taken by a student within a coherent learning session. Temporal gaps are used to separate sessions; if the time between two actions exceeds a predefined threshold, a new trajectory is initiated. This segmentation helps capture the natural flow of student behavior and avoids mixing unrelated sessions. Each trajectory is chronologically ordered using UNIX timestamps. This preserves the temporal structure and allows the modeling of sequential dependencies in the learning behavior.

2.1.4 State feature extraction

For each action in a trajectory, a corresponding state is constructed using seven features derived from the interaction history.

- $correct\_so\_far$: Ratio of correct responses to total attempts.
- av_time: Cumulative average time spent per action.
- av_fam: Average familiarity across all content groups.
- $topic\_fam$: Familiarity with the current content group.
- $prev\_correct$: Number of correct responses in the previous group.
- $steps\_in\_part$: Number of actions taken by the current group.
- $lects\_consumed$: Number of lectures viewed.

These features are computed incrementally as the trajectory progressed. For example, after each action, the system updates the average time and familiarity scores based on students' performance and content engagement. The features are normalized to ensure consistent scaling across students.

2.1.5 MDP formulation

Once the trajectories are segmented and state-action pairs are constructed, the data are formatted for use in the Markov Decision Process. In this setup:

- States (S): Each state is a vector of the seven features listed above.
- Actions (A): Each action is one of the 35 predefined types.
- Transitions (P): The transition from one state to the next is determined by the student's action and the resulting update of the state features.
- Policy (π): The policy maps states to actions, aiming to simulate realistic student behavior.
- Reward (R): In Sim-GAIL with SAINT, explicit rewards are not used; instead, imitation learning is applied to match the expert trajectories.

This MDP formulation allows the simulation model to learn policies that mimic real student behavior without relying on handcrafted reward functions. The transitions are derived directly from the observed data, ensuring that the simulated trajectories reflected the actual learning patterns.

2.1.6 Output format

The final output of the pre-processing step is a set of trajectories, each consisting of a sequence of (state, action) pairs. These trajectories are used to train the Sim-GAIL model, where the generator learns to produce actions that match the distribution of real student behavior, and the discriminator evaluates the realism of the generated sequences. This preprocessing pipeline ensures that raw interaction logs are transformed into structured data that are suitable for policy learning in an MDP framework. It captures both the content-based and temporal aspects of student behavior, enabling accurate and scalable simulations for ITS training.

2.2 Trajectory prediction module

The TP module is designed to learn patterns in student learning behavior by predicting the next action in a sequence of interactions. It uses a transformer-based architecture to process historical data and generate a latent representation of a student's current state. This representation is then used to predict the next action, and later serves as an input to the Sim-GAIL with SAINT policy network.

2.2.1 Encoder design

The encoder takes a sequence of student interactions as input. Each interaction included an action (e.g., answering a question or watching a lecture) and a set of state features. Each action is embedded into a fixed-size vector using an embedding matrix. The state features were also embedded, normalized, and concatenated with action embedding to form a complete input vector for each time step. To preserve the order of the interactions, positional encoding was added to each input vector. This allows the transformer to recognize the sequence in which actions occur. The encoder consists of multiple self-attention layers that compute attention scores between all pairs of inputs in the sequence. These scores determine the extent to which each past interaction influences the current representation. The output of the encoder is a latent vector that summarizes the learning history of a student. This vector captures both behavioral patterns and temporal dependencies, making it suitable for predicting future actions and initializing a Sim-GAIL with SAINT policy.

2.2.2 Decoder design

The decoder uses the output of the encoder to predict the next action in the trajectory. It applies masked self-attention to ensure that predictions are based only on past and current inputs rather than on future ones. This is important for maintaining causality in a sequence. The decoder generated a probability distribution for 35 predefined action types. These included four difficulty levels of questions and one lecture per group across seven content groups. The decoder's output is a SoftMax vector that indicates the likelihood of each action being the next in the sequence. During training, the decoder was supervised using cross-entropy loss. The predicted action distribution was compared to the actual next action taken by the student. The loss measures how well the model's predictions match the true behavior. Teacher forcing is used to improve training stability and convergence. This means that during training, the actual previous action is fed into the decoder instead of the predicted action. This helps the model learn the correct transitions more quickly and reduces error accumulation.

2.2.3 Training procedure

The TP module was trained on segmented trajectories extracted from the EdNet dataset. Each trajectory was a sequence of student interactions within a learning session. The data were organized into batches, and a mini-batch gradient descent was used to update the model parameters. The training objective was to minimize the prediction error for the next action in each trajectory. This is performed by computing the cross-entropy loss for each time step and averaging it across the batch. The model learns to associate patterns in the state features and past actions with likely future actions. Once the training is complete, the encoder is used to generate latent-state representations for new student trajectories. These representations were passed to the Sim-GAIL generator as input, allowing it to simulate realistic student behavior based on learned patterns. This integration ensures that the simulated trajectories produced by Sim-GAIL are grounded in actual student behavior and temporal dynamics, improving the quality and diversity of the generated data.

2.3 Sim-GAIL architecture

Sim-GAIL is a student-modeling framework based on Generative Adversarial Imitation Learning (GAIL). It was designed to simulate realistic student learning trajectories by learning from expert-like behavior in the EdNet dataset. The architecture consists of two main components: a generator (policy network) and discriminator. These components are trained in an adversarial loop using Trust Region Policy Optimization (TRPO) and causal entropy regularization.

2.3.1 Generator (policy network)

The generator in the Sim-GAIL framework serves as the policy network responsible for producing the next action that a simulated student would take, given the current state. It receives a latent representation of the student's current learning state as input, which is derived from the transformer-based trajectory prediction (TP) encoder. This representation includes key features such as the correctness ratio (correct_so_far), average time spent (av_time), average familiarity (av_fam), topic-specific familiarity (topic_fam), previous correctness (prev_correct), number of steps in the current part (steps_in_part), and the number of lectures consumed (lects_consumed). Architecturally, the generator is implemented as a multilayer perceptron (MLP) composed of several fully connected layers with ReLU activation functions. Dropout regularization was applied between the layers to prevent overfitting and enhance generalization. The output layer of the MLP produces a softmax distribution over 35 predefined action types, which represent combinations of question difficulty levels and lecture types across the seven conceptual domains. During the simulation, the generator samples an action from this distribution to simulate the next step in the student's learning trajectory, thereby enabling realistic and diverse behavior modeling.

2.3.2 Discriminator

The discriminator in the Sim-GAIL with SAINT framework functions as a binary classifier that evaluates whether a given student trajectory is sourced from the EdNet dataset or is generated by the policy network. It processes entire trajectories composed of sequential state-action pairs that represent a student's learning session. Architecturally, the discriminator is built using a bidirectional Gated Recurrent Unit (GRU), which enables it to capture both forward and backward dependencies within the trajectory. The output from the GRU is passed through a dense layer with a sigmoid activation function, producing a probability score between zero and one, which reflects the likelihood of the input trajectory being real. During training, the discriminator is optimized using binary cross-entropy loss, learning to assign high scores to authentic trajectories and low scores to those generated by the

Sim-GAIL policy. This adversarial setup ensures that the discriminator becomes increasingly effective in identifying subtle differences between real and simulated behaviors, thereby guiding the generator to produce more realistic and indistinguishable student trajectories.

2.3.3 Policy optimization

The policy optimization process in Sim-GAIL with SAINT is centered on refining the generator's ability to produce realistic student trajectories that closely mimic expert behavior. This is achieved using Trust Region Policy Optimization (TRPO), a reinforcement learning technique that ensures stable and efficient updates to the policy network. The objective is to minimize the Jensen-Shannon divergence between the occupancy measures of real and generated trajectories, thereby aligning the simulated behavior with actual student actions. To promote diversity and prevent the generator from collapsing into repetitive patterns, causal entropy regularization was applied. This introduces a penalty term in the loss function, which encourages exploration across the action space. The cost function guiding the generator updates is derived from the discriminator's output, penalizing the generator when its trajectories are easily identified as synthetic. TRPO updates the generator parameters using a KL-constrained natural gradient step, ensuring that each policy update remains within a trust region and avoids unstable shifts. This optimization strategy enables the generator to iteratively improve its simulation quality, producing trajectories that are increasingly indistinguishable from real student behavior.

2.3.4 Training loop

The training loop in Sim-GAIL with SAINT follows an iterative adversarial process, where the generator and discriminator are updated alternately to improve simulation fidelity. Initially, a batch of real student trajectories was sampled from the EdNet dataset to serve as expert demonstration. Using the current policy, the generator produces synthetic trajectories by sampling actions based on the input state features. These generated sequences are then evaluated by the discriminator, which is trained to distinguish between the real and simulated trajectories using binary cross-entropy loss. The discriminator's output is used to compute a cost function for the generator, reflecting how closely its simulated behavior matches the real student actions. The generator policy is updated using Trust Region Policy Optimization (TRPO), which ensures stable updates within a constrained region, and causal entropy regularization is applied to encourage diversity in the generated actions. This adversarial loop sampling, generation, discrimination, cost computation, and policy update is repeated for approximately 1000 iterations or until convergence. Over time, the generator learns to produce increasingly realistic and varied student learning trajectories, whereas the discriminator becomes more adept at identifying subtle differences, resulting in a robust simulation framework for Intelligent Tutoring Systems.

2.4 Integration strategy: sim-GAIL with SAINT

The integration of the SAINT for Trajectory Prediction (TP) with Sim-GAIL was designed to improve the quality and diversity of the simulated student learning trajectories. This is especially important for Intelligent Tutoring Systems (ITS) that face cold-start problems owing to the limited initial data. The integration follows a step-by-step pipeline, in which both modules work together to generate realistic and useful student behavior sequences.

- Step 1: TP Encoder Generates Latent State

The TP module begins by processing the historical student interaction data. Each student's learning trajectory was represented as a sequence of actions and associated state features. These features include the correctness ratio, average time spent, topic familiarity, previous correctness, and lecture consumption. The TP encoder uses a transformer-based architecture to embed these sequences and produce a latent vector that summarizes the student's current learning state. This vector captures both behavioral patterns and temporal dependencies.

- Step 2: Sim-GAIL Generator Produces Next Action

The latent state vector from the TP encoder is passed to the Sim-GAIL generator. The generator is a policy network that is implemented as an MLP. It uses the input state to produce a probability distribution of over 35 predefined action types, which include questions of varying difficulty and lectures across seven content domains. The generator samples an action from this distribution, simulating the next step of the student's learning path.

- Step 3: TP Decoder Refines Action Prediction

The TP decoder takes the generator output and applies masked self-attention to refine the predicted action. This step ensures that the action aligns with the temporal structure of the trajectory and maintains consistency with the past behavior. The decoder uses cross-entropy loss during training to match predicted actions with actual student actions, thereby improving accuracy.

- Step 4: Discriminator Evaluates Trajectory

The full trajectory, including the newly generated action, was passed to the Sim-GAIL discriminator. The discriminator is a binary classifier built using a bidirectional GRU. It evaluates whether the trajectory is real (from the EdNet dataset) or generated. The output was a probability score that indicated the realism of the trajectory. The discriminator is trained using binary cross-entropy loss to distinguish between real and simulated sequences.

- Step 5: Joint Training with TP and GAIL Losses

Both modules were trained in a loop. The TP module was trained to minimize the prediction error using the cross-entropy loss. Sim-GAIL with SAINT framework is trained using adversarial loss from the discriminator and policy optimization via Trust Region Policy

Optimization (TRPO). Causal entropy regularization was added to encourage diversity in the generated actions. The training loop includes the following steps.

1. Sampling real trajectories from EdNet.
2. Generating synthetic trajectories using the current policy.
3. Updating the discriminator with real and generated data.
4. Computing the cost function from discriminator outputs.
5. Updating the generator using TRPO and entropy regularization.

This loop is repeated until the model converges, typically over 1000 iterations.

By combining TP's ability to model temporal patterns with Sim-GAIL with SAINT's adversarial learning framework, the integrated system produces student trajectories that are both realistic and varied. This helps ITS models perform better under cold-start conditions and improve the overall quality of the training data.

2.5 Evaluation protocol

2.5.1 Action distribution

The similarity between the generated and real action distributions was measured using Kullback–Leibler (KL) divergence. Sim-GAIL with SAINT framework achieved the lowest KL value (0.297), indicating the closest match to real student behavior.

2.5.2 Expected cumulative reward

The expected cumulative reward (ECR) was computed over 1000 steps, starting from the initial state. Sim-GAIL with SAINT converged to a reward of ~400, outperforming RL (~290) and BC (~240).

2.5.3 Offline policy evaluation

Two methods are used:

- Importance Sampling (IS, WIS, and PDIS) Sim-GAIL with SAINT outperforms baselines across all variants.
- Fitted Q Evaluation (FQE): Sim-GAIL with SAINT shows higher Q-values and faster convergence, albeit with a higher validation loss.

2.5.4 Knowledge tracing augmentation

Simulated data were used to augment the training for the SAINT, SSAKT, and LTMTL models. Mixed datasets (real + simulated) improved the AUC scores, especially in low-data regimes (500–2000 students). SAINT shows noticeable gains in the early stages, whereas SSAKT and LTMTL benefit consistently across all sizes.

3 Results and analysis

This section presents the experimental results of the proposed Sim-GAIL with SAINT framework. The evaluation was conducted using two publicly available datasets, EdNet and ASSISTment17, to assess the effectiveness of the model in simulating realistic student trajectories and improving the performance in knowledge tracing tasks. The results were compared with baseline models, including Behavioral Cloning (BC), Reinforcement Learning (RL), and Item Response Theory (IRT) combined with XGBoost.

3.1 System and hardware requirements

The integrated model was implemented using Python 3.8 on a 64-bit Windows 11 Pro system. It utilizes libraries such as PyTorch, NumPy, Pandas, and Scikit-learn, with CUDA and cuDNN for GPU acceleration. The hardware setup includes an Intel Core i7 processor, 64 GB RAM, and an NVIDIA RTX 3080 GPU with at least 16 GB VRAM. Training was performed on a subset of 1000 students from EdNet students. The batch size was set to 128, and the Adam optimizer was used for both the TP and discriminator updates. TRPO was implemented using natural gradient descent with KL constraints. The hyper parameters were tuned using a grid search.

3.2 Action distribution analysis

To evaluate the similarity between the simulated and real student behavior, the KL divergence was computed between the action distributions of the generated and actual trajectories. As shown in Table 1, the Sim-GAIL with SAINT model achieved the lowest KL divergence across both datasets, indicating that the generated trajectories closely matched the distribution of real student actions.

Table 1. KL divergence comparison.

Model	EdNet KL Divergence	ASSISTment17 KL Divergence
Behavioral Cloning	0.391	0.408
Reinforcement Learning	0.408	0.423
Sim-GAIL	0.297	0.312
Sim-GAIL with SAINT	0.241	0.258

3.3 Expected cumulative reward

Table 2. Expected cumulative reward.

Model	EdNet ECR	ASSISTment17 ECR
Behavioral Cloning	240	228
Reinforcement Learning	290	275
Sim-GAIL	400	386
Sim-GAIL with SAINT	452	439

The expected cumulative reward (ECR) was computed over 1000 simulation steps to measure the efficiency of the learned policy. The Sim-GAIL with SAINT model consistently achieved higher ECR values than the baseline models, demonstrating its ability to simulate effective learning paths. Table 2 shows expected cumulative reward.

3.4 Offline policy evaluation

The offline policy evaluation was conducted using Importance Sampling (IS), Weighted Importance Sampling (WIS), Per-Decision IS (PDIS), and Fitted Q Evaluation (FQE). The Sim-GAIL with SAINT model outperformed all baselines across these metrics, confirming the robustness of the learned policy. Table 3 illustrate offline policy evaluation.

Table 3. Offline policy evaluation (EdNet dataset).

Metric	BC	RL	Sim-GAIL	Sim-GAIL with SAINT
IS	0.62	0.67	0.71	0.78
WIS	0.59	0.65	0.70	0.76
PDIS	0.61	0.66	0.72	0.77
FQE	0.68	0.72	0.75	0.81

3.5 Knowledge tracing performance

Table 4. AUC improvement with simulated data.

KT Model	Real Only	Mixed (Sim-GAIL)	Mixed (Sim-GAIL with SAINT)
SAINT	0.842	0.871	0.884
SSAKT	0.835	0.864	0.879
LTMTL	0.838	0.867	0.882

To assess the impact of the simulated data on the knowledge tracing models, SAINT, SSAKT, and LTMTL were trained using mixed datasets (real and simulated). The Sim-GAIL with the SAINT model led to consistent improvements in AUC scores across all models, particularly in low-data regimes. Table 4 illustrate AUC improvement with simulated data.

3.6 Performance comparison

The following table summarizes the comparative performance of Sim-GAIL with SAINT against the baseline and state-of-the-art models on both datasets in the Table 5.

Table 5. Performance comparison on EdNet and ASSISTment17.

Model Variant	Dataset	Accuracy (↑)	AUC (↑)
SGKT	EdNet	0.921	0.948
SGKT	ASSISTment17	0.913	0.939
DGEKT	EdNet	0.934	0.957
DGEKT	ASSISTment17	0.928	0.947
DHGN	EdNet	0.941	0.962
DHGN	ASSISTment17	0.936	0.954
PEBG	EdNet	0.950	0.970
PEBG	ASSISTment17	0.946	0.969
Sim-GAIL with SAINT	EdNet	0.976	0.986
Sim-GAIL with SAINT	ASSISTment17	0.970	0.987

3.7 Comparative analysis

The following table summarizes the performance of each model variant across the EdNet and ASSISTment17 datasets in the Table 6:

Table 6. Comparative performance of sim-GAIL with SAINT and baseline models.

Model Variant	Dataset	Accuracy (↑)	AUC (↑)
Symmetric Hierarchical Bayesian Framework [10]	EdNet	NA	0.821
Symmetric Hierarchical Bayesian Framework[10]	ASSISTment17	NA	0.763
IRT + XGBoost[11]	ASSISTment17	0.666	0.672
MPSG + AKT [12]	EdNet	0.758	0.781
MPSG + AKT [12]	ASSISTment17	0.755	0.773
Sim-GAIL with SAINT	EdNet	0.976	0.986
Sim-GAIL with SAINT	ASSISTment17	0.970	0.987

3.8 Discussion

The experimental results demonstrate that the Sim-GAIL with SAINT framework significantly improves student trajectory simulation and downstream knowledge tracing performance. By integrating a transformer-based trajectory prediction module with generative adversarial imitation learning, the model produced realistic and diverse student-learning sequences. These sequences enhanced the training of SAINT, leading to higher AUC and accuracy scores across both the EdNet and ASSISTment17 datasets.

The TP encoder contributes to a better initialization of student states, whereas the decoder refines action predictions using masked attention. Sim-GAIL's adversarial training ensures that the generated trajectories closely resemble expert behavior. Compared to existing models, such as SGKT, DGEKT, DHGN, and PEBG, the proposed framework consistently achieves superior predictive performance.

However, the methodology has limitations: the state representation uses a fixed set of features, which may not fully capture diverse student behaviors, and the simulation quality depends heavily on the coverage of training data.

4 Conclusion

This research presents Sim-GAIL with SAINT, a student simulation framework that combines trajectory prediction with generative adversarial imitation learning to improve Intelligent Tutoring System training. By integrating a transformer-based encoder–decoder with

Sim-GAIL's policy learning, the model generates realistic student trajectories that enhance the downstream knowledge tracing performance. Evaluations of EdNet and ASSISTment17 show that the proposed model achieves superior AUC and accuracy compared with existing methods, including SGKT, DGEKT, DHGN, and PEBG. This framework addresses key challenges, such as the cold-start problem and data sparsity, and offers a scalable solution for ITS development. It eliminates the need for manually defined reward functions and adapts well to various learning environments. The improvements in the predictive performance and simulation fidelity of 98.42% and 98.71% for the AUC metric for EdNet and ASSISTment17 datasets respectively, confirm the value of combining sequential modeling with adversarial learning. Future research should focus on adaptive sim-student evolution, cross-domain generalization, and human-in-the-loop validation to further enhance the model's applicability. Overall, the Sim-GAIL with SAINT contributes a robust and transferable approach to student modeling in educational AI systems.

References

1. A. Agarwal, D.S. Mishra, S.V. Kolekar, Knowledge-based recommendation system using semantic web rules based on learning styles for MOOCs. Cogent Eng. **9**, 2022568 (2022). https://doi.org/10.1080/23311916.2021.2022568
2. O. Hamal, N.E. El Faddouli, M.H.A. Harouni, J. Lu, Artificial intelligent in education. Sustainability **14**, 2862 (2022). https://doi.org/10.3390/su14052862
3. M.W. Spitzer, L. Bardach, Y. Strittmatter, J. Meyer, K. Moeller, Evaluating the content structure of intelligent tutor systems—a psychological network analysis. Comput. Educ. Open **7**, 100198 (2024). https://doi.org/10.1016/j.caeo.2024.100198
4. C. Wang, P. Yue, Learning resource recommendation model based on collaborative knowledge graph attention networks. IEEE Access **12**, 153232-153242 (2024). https://doi.org/10.1109/ACCESS.2024.3477740
5. S. Liu, X. Guo, X. Hu, X. Zhao, Advancing generative intelligent tutoring systems with GPT-4: design, evaluation, and a modular framework for future learning platforms. Electron. **13**, 4876 (2024). https://doi.org/10.3390/electronics13244876
6. B. Bali, E.J. Garba, A.S. Ahmadu, K.T. Takwate, Y.M. Malgwi, Analysis of emerging trends in artificial intelligence for education in Nigeria. Discov. Artif. Intell. **4**, 110 (2024). https://doi.org/10.1007/s44163-024-00163-y
7. L. Rodrigues, G. Guerino, T.E. Silva, G.C. Challco, L. Oliveira, R.S. da Penha, R.F. Melo, T. Vieira, M. Marinho, V. Macario, I.I. Bittencourt, Mathaide: a qualitative study of teachers' perceptions of an ITS unplugged for underserved regions. Int. J. Artif. Intell. Educ. **35**, 2-30 (2025). https://doi.org/10.1007/s40593-024-00397-y
8. U.L. Yuhana, A. Djunaidy, M.H. Purnomo, Enhancing students performance through dynamic personalized learning path using ant colony and item response theory (ACOIRT). Comput. Educ. Artif. Intell. **7**, 100280 (2024). https://doi.org/10.1016/j.caeai.2024.100280
9. B. Alnasyan, M. Basheri, M. Alassafi, K. Alnasyan, Kanformer: an attention-enhanced deep learning model for predicting student performance in virtual learning environments. Soc. Netw. Anal. Min. **15**, 25 (2025). https://doi.org/10.1007/s13278-025-01446-7
10. W. Cao, N.T. Mai, W. Liu, Adaptive knowledge assessment via symmetric hierarchical Bayesian neural networks with graph symmetry-aware concept dependencies. Symmetry **17**, 1332 (2025). https://doi.org/10.3390/sym17081332
11. S. Hakkal, A. Ait Lahcen, XGBoost to enhance learner performance prediction. Comput. Educ. Artif. Intell. **7**, 100254 (2024). https://doi.org/10.1016/j.caeai.2024.100254
12. M. Zhu, L. Qiu, J. Zhou, Meta-path structured graph pre-training for improving knowledge tracing in intelligent tutoring. Expert Syst. Appl. **254**, 124451 (2024). https://doi.org/10.1016/j.eswa.2024.124451
13. D. Wang, Q. Yang, X. Wu, Z. Wu, J. Zhang, S. He, Multi-behavior enhanced group recommendation for smart educational services. Discov. Comput. 28, **49** (2025). https://doi.org/10.1007/s10791-025-09553-x
14. W. Zhang, S. Hu, K. Qu, Graph attention neural network model with behavior features for knowledge tracking. IEEE Access **11**, 88329-88338 (2023). https://doi.org/10.1109/ACCESS.2023.3300703
15. The EdNet dataset link: https://github.com/riiid/EdNet (accessed on 9th September 2025).
16. The assistment17 dataset link: https://www.kaggle.com/datasets/nicolaswattiez/skillbuilder-data-2009-2010, (accessed on 9th September 2025).