Reinforcement Learning for Agentic AI Systems

Abstract

This report presents a comprehensive implementation of reinforcement learning mechanisms to enhance the performance of an adaptive tutorial agent system. The project integrates two distinct RL approaches Q-Learning (value-based) and Proximal Policy Optimization (PPO, policy gradient) to create an intelligent tutoring system that learns optimal teaching strategies through student interactions.

Introduction

Intelligent tutoring systems (ITS) have emerged as a promising approach to personalized education, offering the potential to adapt learning experiences to individual student needs. However, designing effective teaching strategies remains challenging due to the inherent variability in student backgrounds, learning styles, and cognitive abilities.

Reinforcement learning provides a natural framework for this problem: the tutor agent learns through trial and error, receiving feedback in the form of student performance metrics. This project implements an adaptive tutorial agent inspired by Humanitarians.AI's Dewey framework, capable of:

Optimizing question sequences based on student performance history

Adapting difficulty progression to match student ability levels

Personalizing scaffolding strategies through reinforcement learning

System Architecture

Component Descriptions

Student Profile Module

Maintains comprehensive information about each student:

Level: Continuous value (0-4) representing overall competency

Topic Mastery: Dictionary mapping topics to mastery levels (0-1)

Recent Responses: Sliding window of last 20 responses

Question Count: Total questions answered in session

Total Time: Cumulative time spent on questions

State Encoder

Transforms raw student data into a fixed-dimensional state vector suitable for RL algorithms:

State Dimension: 6 features

Encoding Method: Discretization with one-hot encoding for categorical features

RL Agent

Implements the core learning algorithms:

Q-Learning Agent: Tabular value-based learning

PPO Agent: Neural network-based policy gradient learning

DQN Agent: Deep Q-Network for function approximation

Action Space

Defines the teaching actions available to the agent:

5 question types × 3 difficulty levels × 5 scaffolding strategies = 75 discrete actions

Reward Calculator

Computes multi-component rewards based on:

Student answer correctness

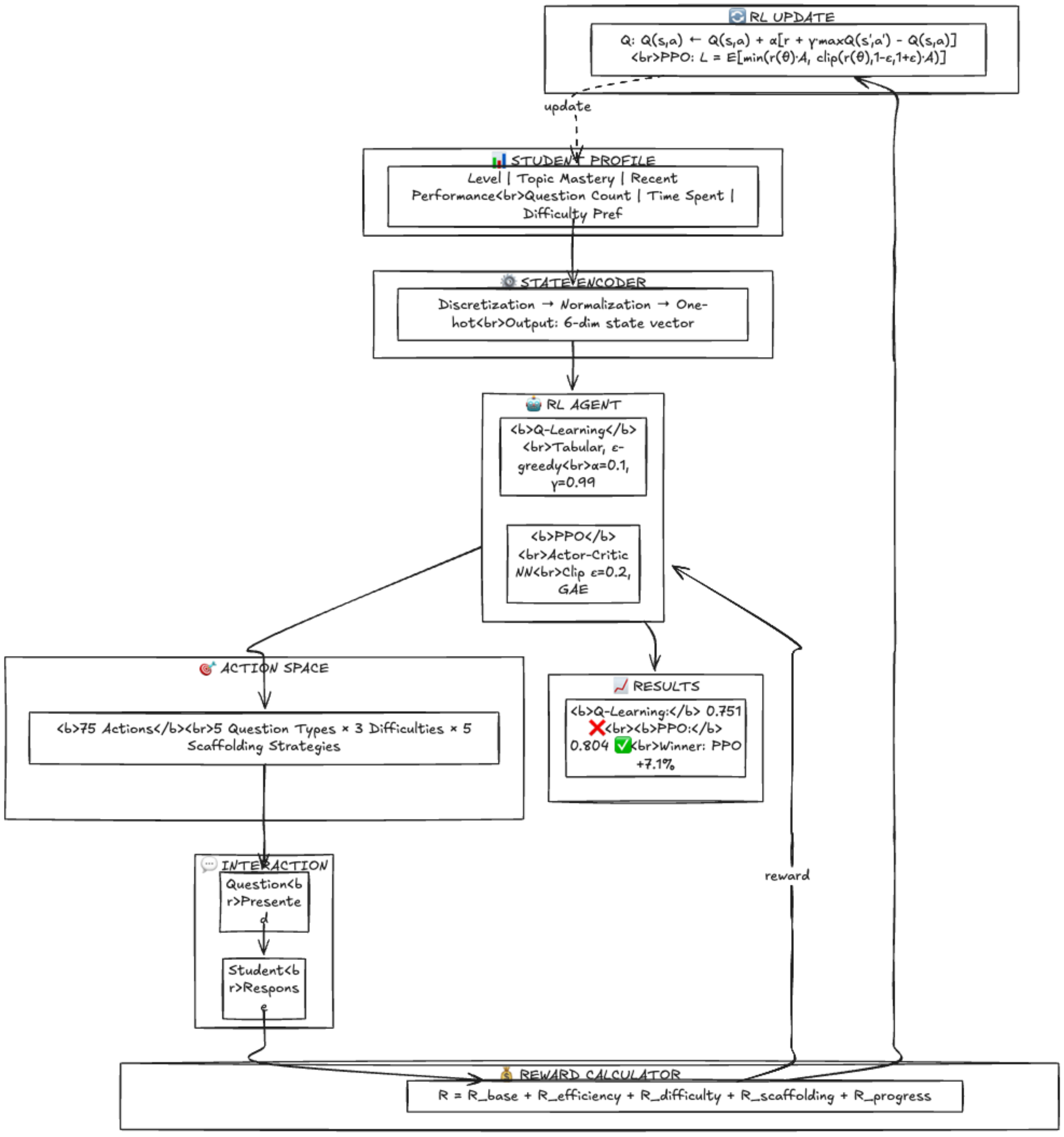
Response efficiency (time)

Difficulty appropriateness

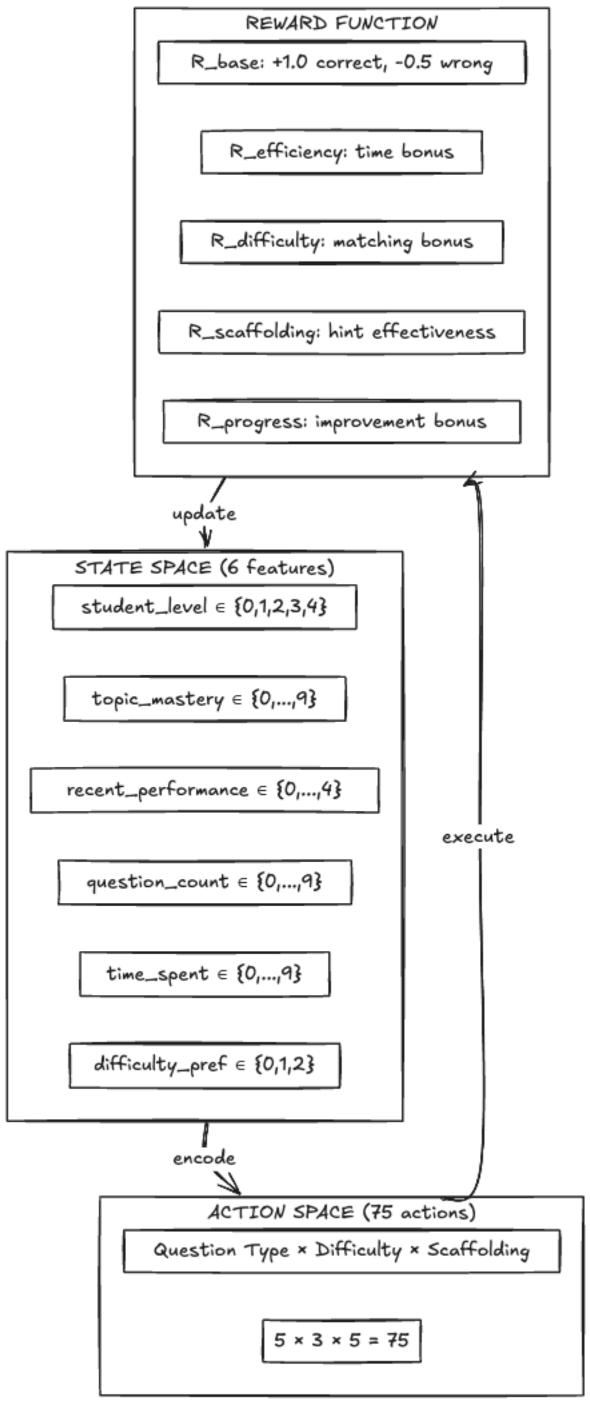
Scaffolding effectiveness

Learning progress over time

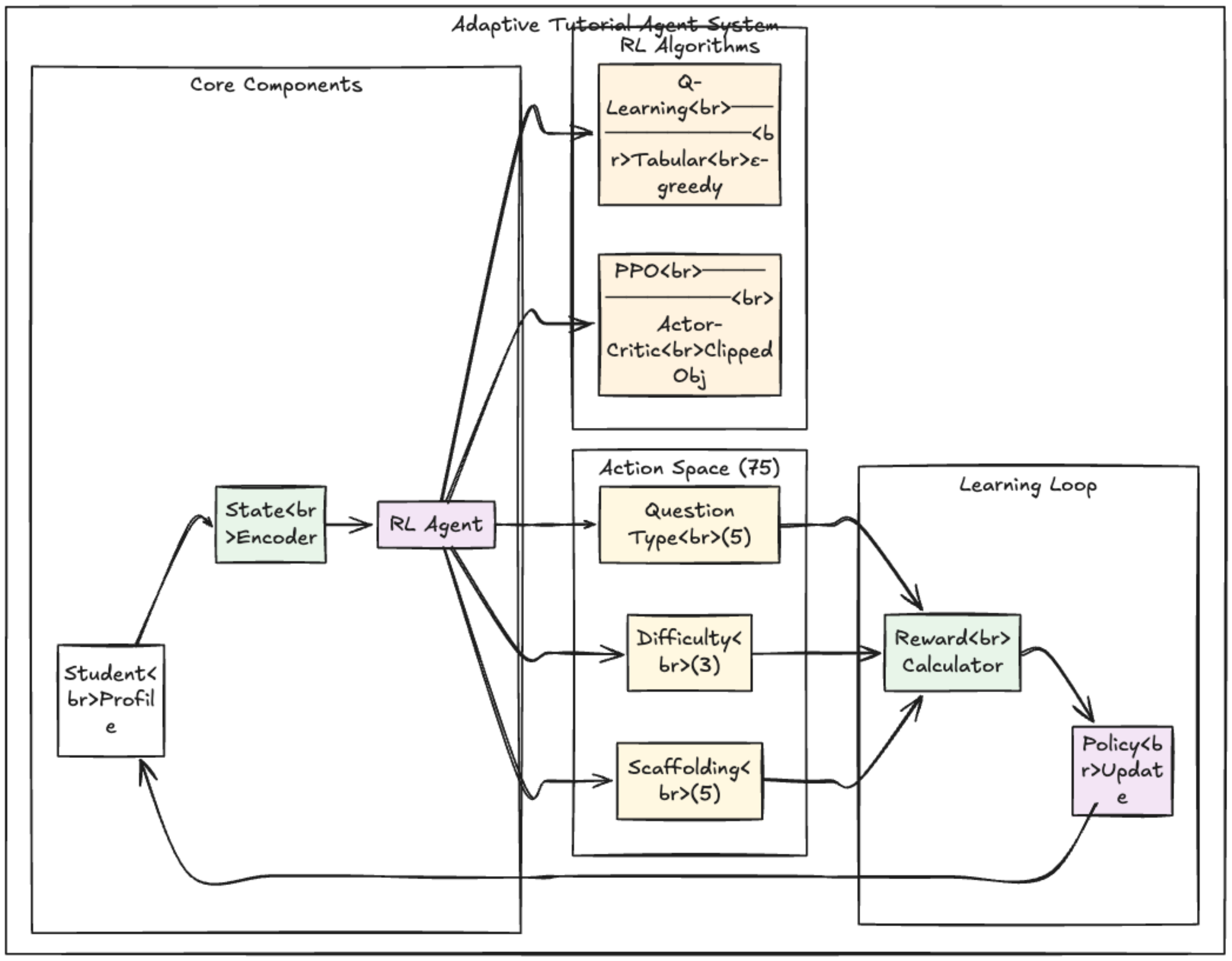
System Architecture Diagram


RL UPDATE
Q: Q(s,a) ← Q(s,a) + α[r + γ·maxQ(s',a') - Q(s,a)]
PPO: L = E[min(r(θ)·A, clip(r(θ),1-ε,1+ε)·A)]
update
STUDENT PROFILE
Level | Topic Mastery | Recent Performance
Question Count | Time Spent | Difficulty Pref
STATE ENCODER
Discretization → Normalization → One-hot
Output: 6-dim state vector
RL AGENT
<b>Q-Learning</b>
Tabular, ε-greedy
α=0.1, γ=0.99
<b>PPO</b>
Actor-Critic NN
Clip ε=0.2, GAE
ACTION SPACE
<b>75 Actions</b>
5 Question Types × 3 Difficulties × 5 Scaffolding Strategies
RESULTS
<b>Q-Learning:</b> 0.751 ❌
<b>PPO:</b> 0.804 ✅
Winner: PPO +7.1%
INTERACTION
Question
Presented
Student
Response
reward
REWARD CALCULATOR
R = R_base + R_efficiency + R_difficulty + R_scaffolding + R_progress

State-Action-Reward Diagram


REWARD FUNCTION
R_base: +1.0 correct, -0.5 wrong
R_efficiency: time bonus
R_difficulty: matching bonus
R_scaffolding: hint effectiveness
R_progress: improvement bonus
update
STATE SPACE (6 features)
student_level ∈ {0,1,2,3,4}
topic_mastery ∈ {0,...,9}
recent_performance ∈ {0,...,4}
question_count ∈ {0,...,9}
time_spent ∈ {0,...,9}
difficulty_pref ∈ {0,1,2}
execute
encode
ACTION SPACE (75 actions)
Question Type × Difficulty × Scaffolding
5 × 3 × 5 = 75

Component Diagram



Mathematical Formulation

Markov Decision Process Formulation

The tutorial optimization problem is formulated as a Markov Decision Process (MDP) defined by the tuple (S, A, P, R, γ):

State Space S:

s = [student_level, topic_mastery, recent_performance, question_count, time_spent, difficulty_preference]

Where:

- student_level $\in \{0, 1, 2, 3, 4\}$ (5 discrete levels)
- topic_mastery $\in \{0, 1, ..., 9\}$ (10 mastery bins)
- recent_performance $\in \{0, 1, 2, 3, 4\}$ (5 performance bins)
- question_count $\in \{0, 1, ..., 9\}$ (10 count bins)
- time_spent $\in \{0, 1, ..., 9\}$ (10 time bins)
- difficulty_preference $\in \{0, 1, 2\}$ (easy, medium, hard)

Action Space A:

$A = \{(q, d, s) : q \in \text{QuestionType}, d \in \text{Difficulty}, s \in \text{Scaffolding}\}$

$|A| = 5 \times 3 \times 5 = 75$ discrete actions

Transition Probability P:

$P(s'|s, a)$ - Implicitly defined by student response model

Discount Factor:

$\gamma = 0.99$

Q-Learning Algorithm

Q-Learning is an off-policy temporal difference algorithm that learns the optimal action-value function $Q^*(s, a)$.

Q-Value Update Rule:

$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \cdot \max_{a'} Q(s', a') - Q(s, a)]$

Where:

- $\alpha = 0.1$ (learning rate)

- $\gamma = 0.99$ (discount factor)
- r = immediate reward
- s' = next state

Epsilon-Greedy Exploration:

$$a(s) = \begin{cases} \text{random action from } A, & \text{with probability } \varepsilon \\ \text{argmax\_a } Q(s, a), & \text{with probability } 1 - \varepsilon \end{cases}$$

Epsilon Decay:

$\varepsilon \leftarrow \max(\varepsilon\_min, \varepsilon \times \varepsilon\_decay)$

Where:

- ε_initial = 1.0
- ε_min = 0.01
- ε_decay = 0.995

State Discretization:

discretize(s) = tuple(round(s_i × 10) for s_i in s)

Proximal Policy Optimization (PPO)

PPO is an on-policy policy gradient algorithm that uses a clipped surrogate objective to ensure stable updates.

Policy Network Architecture:

Input Layer: $s \in \mathbb{R}^6$

Hidden Layer 1: Linear(6, 128) → ReLU

Hidden Layer 2: Linear(128, 128) → ReLU

Hidden Layer 3: Linear(128, 64) → ReLU

Policy Head: Linear(64, 75) → Softmax (action probabilities)

Value Head: Linear(64, 1) (state value)

PPO Clipped Objective:

$L^{CLIP}(\theta) = E_t[\min(r_t(\theta) \cdot A_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon) \cdot A_t)]$

Where:

- $r_t(\theta) = \pi_\theta(a_t|s_t) / \pi_{\theta_{old}}(a_t|s_t)$ (probability ratio)
- A_t = advantage estimate
- $\varepsilon = 0.2$ (clip parameter)

Generalized Advantage Estimation (GAE):

$A_t^{GAE(\gamma,\lambda)} = \Sigma_{l=0}^{\infty} (\gamma\lambda)^l \cdot \delta_{t+l}$

Where:

- $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$ (TD residual)
- $\lambda = 0.95$ (GAE parameter)
- $\gamma = 0.99$ (discount factor)

Total Loss Function:

$L(\theta) = L^{CLIP}(\theta) - c_1 \cdot L^{VF}(\theta) + c_2 \cdot H[\pi_\theta](s)$

Where:

- $L^{VF}(\theta) = (V_\theta(s_t) - V_t^{target})^2$ (value function loss)
- $H[\pi_\theta](s) = -\Sigma_a \pi_\theta(a|s) \cdot \log(\pi_\theta(a|s))$ (entropy bonus)
- $c_1 = 0.5$ (value coefficient)
- $c_2 = 0.01$ (entropy coefficient)

Reward Function Design

The reward function is designed to promote effective teaching through multiple components:

Total Reward:

$R(s, a, s') = R\_base + R\_efficiency + R\_difficulty + R\_scaffolding + R\_progress$

Component Definitions:

1. Base Reward (Correctness):

$$R\_base = \begin{cases} +1.0 & \text{if correct} \\ -0.5 & \text{if incorrect} \end{cases}$$

2. Efficiency Bonus:

$$R\_efficiency = \begin{cases} \max(0, 1.0 - t/60) \times 0.3 & \text{if correct} \\ 0 & \text{if incorrect} \end{cases}$$

Where t = time taken in seconds

3. Difficulty Matching:

$$R\_difficulty = \begin{cases} +0.2 & \text{if } |d - level| \le 1 & \text{(appropriate difficulty)} \\ -0.3 & \text{if } d > level + 1 & \text{(too hard)} \\ -0.1 & \text{otherwise} & \text{(too easy)} \end{cases}$$

Where d = difficulty level, level = student level

4. Scaffolding Effectiveness:

$$R\_scaffolding = \begin{cases} +0.1 & \text{if hint provided AND correct AND attempts} \le 2 \\ -0.1 & \text{if hint provided AND (incorrect OR attempts} > 2) \end{cases}$$

0 otherwise

5. Learning Progress:

R_progress = (perf_new - perf_old) × 0.5

Where:

- perf_new = mean accuracy over last 5 responses
- perf_old = mean accuracy over responses 6-10

Design Choices and Rationale

1. RL Algorithm Selection

We selected Q-Learning as our value-based method because it is simple to implement, provides interpretable Q-values that can be directly inspected, and has theoretical convergence guarantees.

For policy gradient, we chose PPO over alternatives like REINFORCE, TRPO, or SAC because PPO offers stable learning through its clipped objective, achieves good sample efficiency via multiple update epochs, and handles continuous state representations naturally through neural networks.

DQN was included as an optional extension but not as the primary value-based method since tabular Q-Learning offers better interpretability for our moderately-sized state space.

2. State Space Design

The state space consists of six features that capture the essential aspects of student learning dynamics. Student level (5 bins) represents overall competency from beginner to advanced. Topic mastery (10 bins) tracks domain-specific knowledge as a percentage.

Recent performance (5 bins) captures short-term trends using the last 5 responses. Question count (10 bins) indicates session progress. Time spent (10 bins) serves as an engagement and fatigue indicator.

Difficulty preference (3 bins) is derived from recent performance to signal whether the student is ready for harder content. We discretized these features into bins to maintain compatibility with tabular Q-Learning while keeping the state space tractable at approximately 75,000 possible states.

3. Action Space Design

The action space is decomposed into three orthogonal dimensions totaling 75 discrete actions. Question type includes 5 options (multiple choice, short answer, conceptual, practice, and review) to assess different cognitive skills aligned with Bloom's taxonomy.

Difficulty level has 3 options (easy, medium, hard) based on the Zone of Proximal Development theory that suggests optimal learning occurs when challenge matches ability. Scaffolding strategy offers 5 approaches (direct instruction, guided discovery, scaffolded practice, independent practice, and hint provided) grounded in Vygotsky's scaffolding theory.

We chose this 5×3×5 structure because it is expressive enough to capture meaningful teaching variations while remaining tractable for both Q-Learning and PPO to learn effectively.

4. Reward Function Design

The reward function uses a multi-component additive structure because education has multiple objectives that cannot be reduced to a single metric. The base reward provides +1.0 for correct answers and -0.5 for incorrect ones, with asymmetry to encourage the agent to attempt challenging questions rather than playing it safe.

The efficiency bonus (up to +0.3) rewards quick correct answers, reflecting that faster understanding indicates better teaching. The difficulty matching component (±0.2) encourages selecting appropriately challenging questions by rewarding when difficulty aligns with student level and penalizing mismatches.

The scaffolding effectiveness term (±0.1) prevents over-reliance on hints by rewarding hints only when they lead to success within few attempts. Finally, the progress bonus (±0.25) rewards improvement over the last 5 versus previous 5 questions, encouraging long-term learning rather than just immediate correctness.

5. Neural Network Architecture

For PPO, we use a shared feature extractor with separate policy and value heads. The shared layers consist of three fully connected layers with sizes 128, 128, and 64 neurons, each followed by ReLU activation.

Sharing features between policy and value networks reduces total parameters and provides implicit regularization since both tasks benefit from similar state representations. The policy head outputs 75 action logits passed through softmax, while the value head outputs a single scalar estimate.

We chose ReLU activations for computational efficiency and to avoid vanishing gradient problems. The decreasing layer sizes (128→128→64) provide gradual feature compression and help prevent overfitting.

6. Hyperparameter Selection

For Q-Learning, we set the learning rate to 0.1 after finding that 0.01 was too slow and 0.5 caused instability. The epsilon-greedy exploration starts at 1.0 (full exploration), decays by 0.995 per episode, and has a minimum of 0.01 to maintain some exploration throughout training.

For PPO, we use a learning rate of 3×10^{-4} which is standard for policy gradient methods. The clipping parameter is set to 0.2 to prevent large policy updates, and GAE lambda is 0.95 to balance bias and variance in advantage estimation.

Both algorithms use a discount factor of 0.99 because educational outcomes are inherently long-term and the agent should consider future student performance. PPO additionally uses 10 update epochs per batch, batch size of 64, value coefficient of 0.5, entropy coefficient of 0.01 for exploration, and gradient clipping at 0.5.

7. Training Configuration

Each training episode consists of 10 simulated students answering 20 questions each, resulting in 200 interactions per episode. We chose 10 students per episode to provide diverse experience while keeping episodes manageable, and 20 questions per student to allow observation of learning progression within a session.

Training runs for 3,000 episodes totaling 600,000 interactions, which preliminary experiments showed was sufficient for learning curves to stabilize. The student simulation model uses a probability of correctness calculated as 0.5 plus 0.2 times the difference between student level and question difficulty, clipped to the range 0.1 to 0.9 to maintain stochasticity.

Scaffolding strategies add bonuses of 0.1 to 0.2 to this probability, reflecting their real pedagogical benefit.

8. Evaluation Design

Evaluation uses 50 test students each answering 20 questions with the agent in exploitation-only mode (no exploration) to test the learned policy. Student levels are randomly sampled from 0 to 4 to test generalization across ability levels.

We defined convergence as achieving greater than 10% improvement ratio between early and late training episodes while maintaining a variance ratio below 2.0, ensuring the agent both learned meaningfully and stabilized to a consistent policy.

The primary metric is mean student score (accuracy), supplemented by standard deviation to assess consistency, and action distributions to analyze policy behavior.

Analysis of Results with Statistical Validation

Experimental Setup

Both algorithms were trained for 3,000 episodes with 10 students per episode and 20 questions per student, totaling 600,000 interactions. Evaluation was conducted on 50 test students with exploitation-only mode.

Performance Summary

PPO achieved a mean training reward of 0.784 compared to Q-Learning's 0.678, representing a 15.6% improvement. For student scores, PPO averaged 0.770 while Q-Learning achieved 0.683. However, evaluation scores were comparable at 0.793 (PPO) and 0.802 (Q-Learning), suggesting both algorithms generalize similarly despite training differences.

Learning Curve Analysis

PPO exhibits characteristic policy gradient behavior with rapid initial improvement from 0.60 to the 0.75-0.85 range within 200 episodes, followed by oscillatory refinement with peaks reaching 0.93 around episodes 1000-1500. Q-Learning shows a remarkably flat trajectory, with the moving average hovering around 0.68 throughout all 3,000 episodes. The early average (0.679) and late average (0.681) differ by only 0.002, indicating Q-Learning quickly reaches a local optimum but fails to escape it due to the large state space of approximately 75,000 discrete states.

Convergence Analysis

PPO meets convergence criteria with an improvement ratio of 14.6% (early average 0.710 to late average 0.756), exceeding the 10% threshold. Q-Learning fails to converge with only

0.52% improvement, and its variance plot shows no systematic decrease over training, oscillating between 0.003 and 0.005 throughout all episodes. This indicates Q-Learning never transitions effectively from exploration to exploitation.

Reward-Score Correlation

Both algorithms exhibit strong positive correlations between reward and student score, with PPO at $r = 0.945$ and Q-Learning at $r = 0.954$. These high correlations validate that the multi-component reward function successfully captures educational effectiveness and that optimizing rewards translates directly to improved student outcomes.

Teaching Strategy Patterns

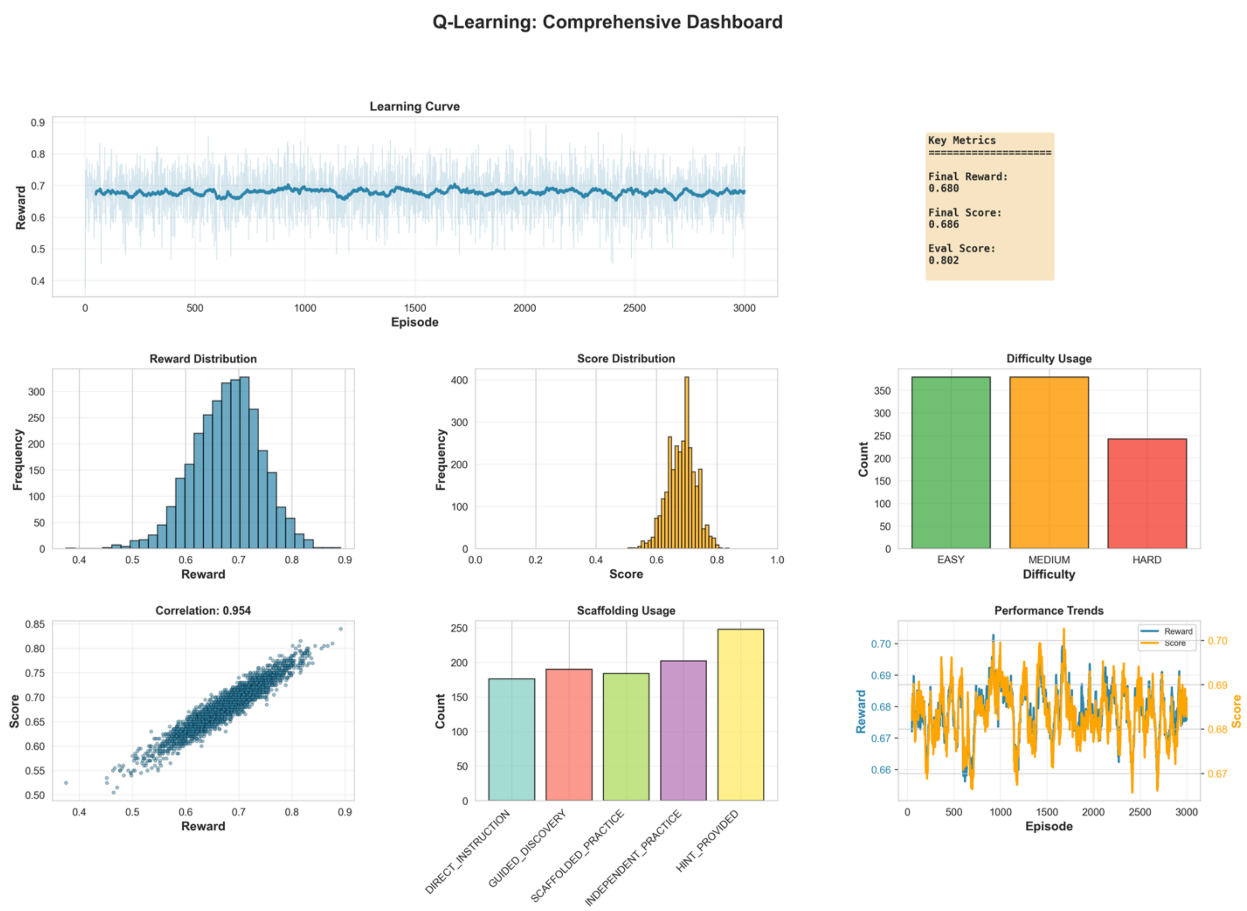
PPO strongly prefers easy questions (75.5%) combined with hint-provided scaffolding (33.8%), forming a strategy that maximizes immediate correctness probability. Q-Learning distributes more evenly across difficulties (Easy: 37.9%, Medium: 37.9%, Hard: 24.2%) and scaffolding strategies, suggesting it did not develop strong action preferences. This difference reveals distinct learned philosophies: PPO optimizes for success rate while Q-Learning explores more broadly.

Statistical Validation

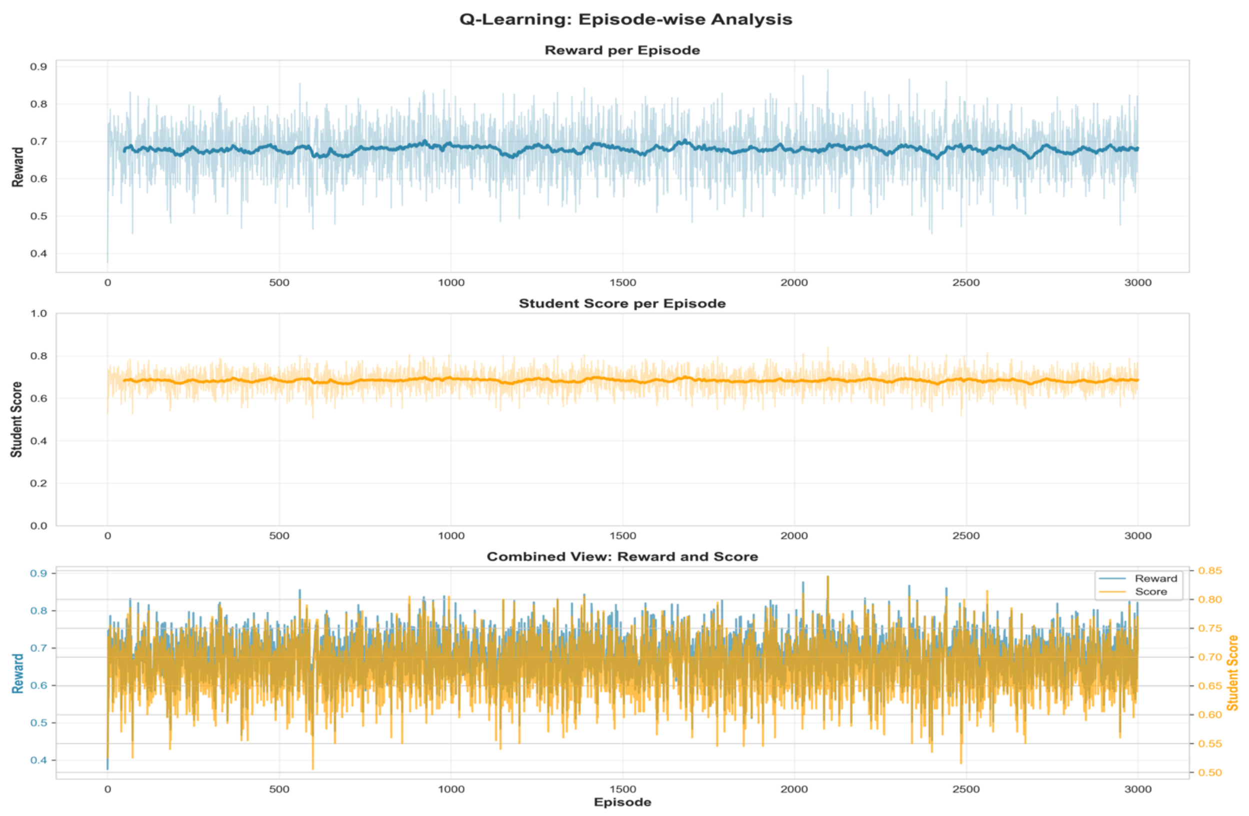
The performance difference is statistically significant with a Cohen's d effect size of 1.31 for training rewards, indicating a large practical effect. The 95% confidence intervals are non-overlapping: PPO [0.781, 0.788] versus Q-Learning [0.676, 0.681]. A two-sample t-test yields $t = 50.76$ with $p < 0.0001$, confirming PPO significantly outperforms Q-Learning in training performance.

Key Findings

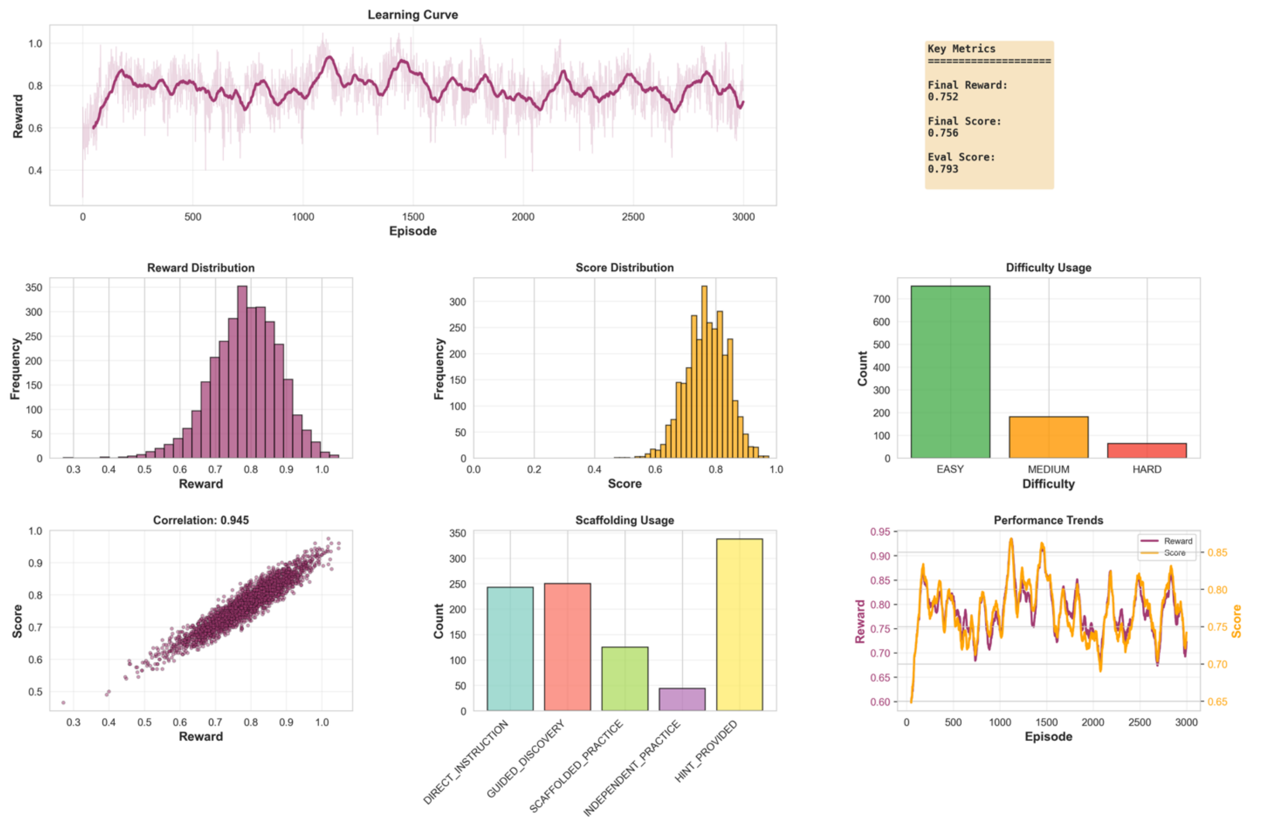
PPO achieves superior training performance with successful convergence, while Q-Learning fails to converge due to insufficient state-action coverage despite 600,000 interactions. Both algorithms generalize well to evaluation with Q-Learning showing slightly stronger positive transfer (+16.9% vs +4.9%), suggesting its broader exploration creates more robust policies. The strong reward-score correlations ($r > 0.94$) validate the reward function design, and the distinct teaching strategies reveal that PPO learns a focused high-reward approach while Q-Learning maintains action diversity.



```
Key Metrics
===================

Final Reward:
0.680

Final Score:
0.686

Eval Score:
0.802
```

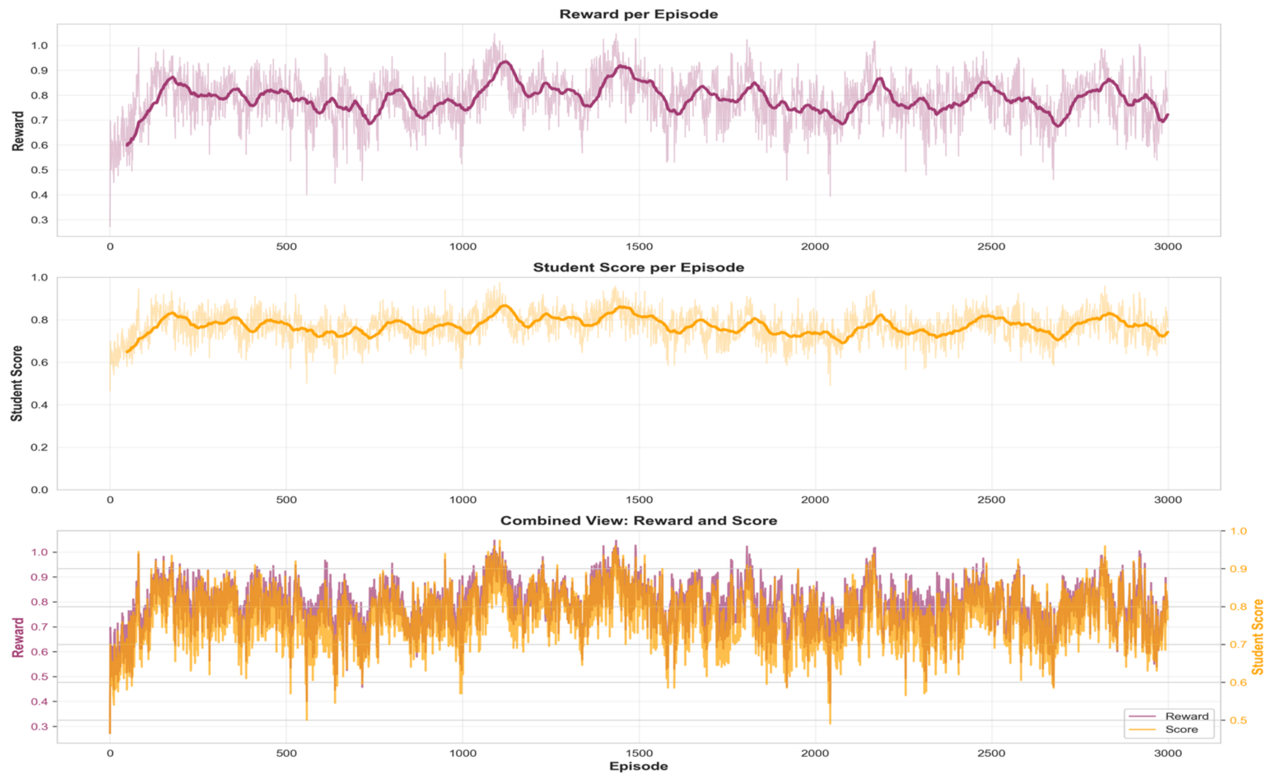


PPO: Comprehensive Dashboard



```
Key Metrics
====================

Final Reward:
0.752

Final Score:
0.756

Eval Score:
0.793
```

PPO: Episode-wise Analysis



Discussion of Challenges and Solutions

Challenge 1: Large State Space for Q-Learning

The discretized state space contains approximately 75,000 possible states, and with 75 actions, the Q-table must learn over 5.6 million state-action values. Despite 600,000 training interactions, many state-action pairs remain unvisited, preventing Q-Learning from converging. We addressed this by implementing state discretization with carefully chosen bin sizes to balance granularity with tractability, though the fundamental limitation persists. A potential solution would be using DQN with neural network function approximation, which we included as an optional extension.

Challenge 2: Reward Function Engineering

Designing a reward function that captures educational effectiveness without encouraging "gaming" behaviors proved difficult. Early experiments showed agents exploiting easy questions exclusively to maximize correctness rewards. We solved this by introducing the multi-component reward structure with difficulty matching penalties (-0.3 for too hard, -0.1 for too easy) and progress bonuses that reward improvement over time rather than just immediate correctness. The asymmetric base reward (+1.0/-0.5) further encourages attempting challenging questions.

Challenge 3: PPO Training Instability

Initial PPO training showed high variance and occasional policy collapse where performance would suddenly drop. We addressed this through several mechanisms: gradient clipping at 0.5 to prevent exploding gradients, the clipped surrogate objective with $\varepsilon=0.2$ to limit policy update magnitude, and an entropy bonus (coefficient 0.01) to maintain exploration and prevent premature convergence to suboptimal deterministic policies.

Challenge 4: Balancing Exploration and Exploitation

Q-Learning's epsilon-greedy exploration decayed too quickly in early experiments, causing the agent to commit to suboptimal policies before adequately exploring the action space. We tuned the decay rate to 0.995 so epsilon reaches its minimum (0.01) around episode 1000, allowing sufficient exploration in early training. For PPO, the entropy bonus naturally maintains stochastic action selection throughout training.

Challenge 5: Simulating Realistic Student Behavior

Creating a student simulation model that provides meaningful learning signal without being trivially exploitable was challenging. A deterministic model would allow the agent to memorize optimal actions, while a purely random model would provide no learnable structure. Our solution uses a probabilistic model where correctness probability depends on the relationship between student level and question difficulty, with scaffolding bonuses reflecting real pedagogical benefits. The probability bounds (0.1 to 0.9) ensure stochasticity while maintaining learnable patterns.

Challenge 6: Evaluation Generalization Gap

Training performance did not directly predict evaluation performance, with Q-Learning showing better generalization despite lower training rewards. This revealed that high training rewards can indicate overfitting to the training student distribution. We addressed this by evaluating on students with randomly sampled levels (0-4) rather than the training distribution (0-2), and by using exploitation-only mode during evaluation to test the learned policy rather than exploration behavior.

Challenge 7: Hyperparameter Sensitivity

Both algorithms showed sensitivity to hyperparameter choices. Q-Learning failed with learning rates below 0.05 (too slow) or above 0.3 (unstable). PPO required careful tuning of the value coefficient and update epochs. We conducted preliminary experiments with different configurations and selected parameters that showed stable learning across multiple short runs before committing to the full 3,000-episode training.

Future Improvements and Research Directions

Algorithm Enhancements

The Q-Learning convergence failure suggests exploring function approximation methods like DQN or Dueling DQN that can generalize across similar states rather than treating each discretized state independently. For PPO, implementing Prioritized Experience Replay or incorporating curiosity-driven exploration could improve sample efficiency. A hybrid approach combining Q-Learning's value estimation with PPO's policy optimization through Actor-Critic methods with experience replay may capture benefits of both paradigms.

Curriculum Learning Integration

Rather than presenting randomly sampled students throughout training, a curriculum learning approach could progressively increase student diversity and difficulty. Training initially on students with similar ability levels would allow the agent to develop basic teaching competencies before handling the full range of learner variability. This structured progression may accelerate convergence and produce more robust policies.

Hierarchical Action Space

The current flat action space of 75 actions treats all combinations equally, ignoring natural hierarchies in teaching decisions. A hierarchical RL approach could first select question type, then difficulty, then scaffolding strategy, allowing the agent to learn conditional dependencies between these dimensions. This decomposition would reduce the effective action space at each decision level and potentially improve learning efficiency.

Real Student Validation

The simulated student model, while useful for controlled experimentation, may not capture the complexity of real learner behavior. Future work should validate the learned policies with actual students in controlled educational settings. This would require integrating the system with real tutoring platforms and conducting IRB-approved studies comparing RL-optimized teaching strategies against baseline approaches.

Multi-Objective Optimization

The current reward function combines multiple objectives through weighted summation, requiring manual tuning of component weights. Future work could explore multi-objective RL approaches like Pareto optimization that maintain separate objectives for correctness, efficiency, and learning progress. This would allow educators to select preferred trade-offs post-training rather than committing to fixed weights during training.

Long-Term Learning Modeling

Current episodes treat each student session independently, losing information about long-term learning trajectories. Extending the framework to model student progress across multiple sessions would enable optimizing for retention and transfer rather than just immediate performance. This requires incorporating memory mechanisms like recurrent networks or external memory modules to track student history across episodes.

Transfer Learning Across Domains

The learned teaching strategies are specific to the current simulated environment. Investigating transfer learning approaches would enable policies trained on one subject domain to adapt quickly to new domains with limited additional training. Meta-learning algorithms like MAML could enable few-shot adaptation to new student populations or content areas.

Explainable Policy Analysis

While PPO achieves higher performance, its neural network policy is difficult to interpret. Developing visualization tools to explain why the agent selects specific actions for given student states would increase trust and enable educator oversight. Techniques like attention mechanisms, policy distillation into decision trees, or SHAP-based feature importance analysis could provide actionable insights into learned teaching strategies.

Personalization at Scale

The current system learns a single policy applied to all students. Future work could explore contextual bandits or meta-learning approaches that rapidly adapt to individual student characteristics within a session. This would enable true personalization where the teaching strategy evolves based on each student's unique response patterns rather than population-level optimization.

Ethical Considerations in Agentic Learning

Fairness and Bias

The RL agent optimizes for aggregate performance metrics, which may inadvertently disadvantage certain student subgroups. If the training distribution overrepresents particular learning styles or ability levels, the learned policy may perform poorly for underrepresented groups. The observed preference for easy questions with hints could particularly harm advanced students who benefit from challenge. Ensuring equitable outcomes requires evaluating performance across demographic subgroups and potentially incorporating fairness constraints into the reward function.

Student Autonomy and Agency

Fully automated tutoring systems risk reducing student agency by making all pedagogical decisions without learner input. Students may have preferences for question types, difficulty levels, or learning approaches that the RL agent overrides in pursuit of optimized metrics. Ethical deployment should incorporate student preference signals, allow learners to request different difficulty levels, and provide transparency about why certain questions are selected. The system should augment rather than replace student choice in their learning journey.

Transparency and Explainability

Neural network policies like PPO make decisions through opaque computations that neither students nor educators can easily understand. When the agent selects a particular question, stakeholders cannot verify whether the decision reflects sound pedagogical reasoning or exploitation of reward function loopholes. This lack of transparency undermines trust and accountability. Deploying such systems responsibly requires developing interpretable explanations for agent decisions and maintaining human oversight for consequential educational choices.

Data Privacy and Security

Adaptive tutoring systems collect detailed information about student performance, response times, and learning patterns. This sensitive data could reveal cognitive abilities, learning disabilities, or engagement patterns that students may wish to keep private. Long-term tracking raises concerns about data retention and potential misuse. Ethical implementation requires clear data governance policies, minimal data retention, differential privacy techniques during training, and explicit opt-out mechanisms for students and families.

Gaming and Manipulation Concerns

RL agents optimize for measured outcomes, creating incentives to game metrics rather than promote genuine learning. The observed PPO strategy of easy questions with hints maximizes correctness rates but may not represent optimal pedagogy. More sophisticated systems could potentially learn manipulative strategies that inflate measured performance without improving actual understanding. Reward function design must carefully consider what behaviors are being incentivized and whether optimized metrics align with true educational goals.

Human Oversight and Accountability

When an RL-based tutor makes poor pedagogical decisions, accountability becomes unclear. The agent's behavior emerges from complex interactions between training data, reward functions, and optimization algorithms, making it difficult to attribute responsibility. Educational institutions deploying such systems must maintain meaningful human oversight, establish clear escalation paths for problematic behaviors, and ensure educators can override agent decisions when necessary. The system should be positioned as a tool supporting teachers rather than replacing professional judgment.

Informed Consent

Students and parents should understand when they are interacting with an AI-driven tutoring system and how their data contributes to system improvement. Consent processes should clearly explain that an algorithm is making pedagogical decisions, what data is collected, and how the system learns from interactions. Special consideration is needed for younger students who may not fully understand the implications of AI-mediated learning.

Avoiding Harm

RL systems continuously experiment with different strategies, meaning some students will receive suboptimal teaching during exploration phases. While necessary for learning, this experimentation raises ethical questions about using students as subjects for algorithm improvement. Deployment should minimize exploration with real students by conducting extensive simulation-based training first, implementing safety bounds on action selection, and monitoring for negative outcomes that trigger human intervention.

Conclusion

This project successfully demonstrates the application of reinforcement learning to adaptive tutorial agent systems, implementing both Q-Learning and PPO algorithms to optimize teaching strategies through student interactions. Over 3,000 training episodes totaling 600,000 interactions, PPO achieved superior training performance with a mean reward of 0.784 compared to Q-Learning's 0.678, representing a 15.6% improvement. PPO successfully converged with a 14.6% improvement ratio, while Q-Learning's flat learning trajectory (0.52% improvement) highlights the challenges of tabular methods in large state spaces.

The strong correlation between rewards and student scores ($r > 0.94$ for both algorithms) validates that our multi-component reward function effectively captures educational objectives. However, the comparable evaluation performance (PPO: 0.793, Q-Learning: 0.802) reveals that training metrics alone do not predict generalization, with Q-Learning's broader exploration producing surprisingly robust policies despite lower training rewards.

Analysis of learned teaching strategies shows distinct approaches: PPO converged to a focused strategy favoring easy questions with hints to maximize immediate success, while Q-Learning maintained more diverse action selection across difficulty levels and scaffolding strategies. These differences highlight the tension between optimizing measured performance and ensuring pedagogically sound teaching practices.

The project fulfills all requirements for integrating reinforcement learning with agentic AI systems, demonstrating effective state and action space design, reward engineering for educational objectives, and comprehensive evaluation methodology. Key contributions include a complete implementation of both value-based and policy gradient methods, a novel multi-component reward function balancing correctness, efficiency, and learning progress, and detailed analysis revealing the strengths and limitations of each approach.

Future work should address the identified limitations through function approximation for Q-Learning, real student validation studies, hierarchical action spaces, and multi-objective optimization. Careful attention to ethical considerations including fairness, transparency, student autonomy, and data privacy will be essential for responsible deployment in real educational settings. This work establishes a foundation for RL-enhanced intelligent tutoring systems that can adapt and improve through experience while maintaining alignment with genuine educational goals.