Abstract

Quiz design is a tedious process that teachers undertake to evaluate the acquisition of knowledge by students. Our goal in this paper is to automate quiz composition from a set of multiple choice questions (MCQs). We formalize a generic sequential decision-making problem with the goal of training an agent to compose a quiz that meets the desired topic coverage and difficulty levels. We investigate DQN, SARSA and A2C/A3C, three reinforcement learning solutions to solve our problem. We run extensive experiments on synthetic and real datasets that study the ability of RL to land on the best quiz. Our results reveal subtle differences in agent behavior and in transfer learning with different data distributions and teacher goals. This was supported by our user study, paving the way for automating various teachers' pedagogical goals.

1 Introduction

Teachers spend considerable time crafting quizzes to evaluate their students' knowledge acquisition [11, 12, 18]. They usually follow a stepwise decision-making process starting from a mix of previously used quizzes and newly designed Multiple-Choice Questions (MCQs). At each step, they seek to ensure that the MCQs forming a quiz cover desired topics and difficulty level distributions. In this paper, our aim is to help teachers automate quiz composition.

Context and motivating example. Quiz creation has recently moved from manual editing to smarter, learning-based systems. Teachers used to spend a lot of time adjusting questions one by one to create a good quiz. Key concerns like topic coverage, difficulty balance, and relevance to the material are still essential for building high-quality quizzes. Consider the six Multi-Choice Questions provided in Figure 1 where the topic, difficulty level, and correct solution of each MCQ are highlighted in bold. A teacher seeking to build a 3-MCQ quiz to test students on two topics, e.g. Statistics and Probabilities, would choose MCQs with varying difficulties, e.g., MCQs 1, 5, and 6 (this answer is not unique). A teacher who aims to vary topics and maintain the same difficulties would choose MCQs 1, 4 and 5, that constitute the only solution in this case. In practice, teachers proceed either by creating MCQs from scratch, or by replacing some MCQs by others until they reach a satisfactory quiz.

To reduce human effort in composing quizzes from large MCQ pools, early systems used rule-based methods with predefined templates and constraints (e.g., "include two geometry questions"), offering limited flexibility and generalization [1, 6]. Later work automated quiz generation using difficulty prediction via lexical and syntactic features [4], or clustering and topic modeling to group or diversify questions [9]. However, these methods target single objectives—such as topic relevance or redundancy reduction—and do not balance multiple pedagogical criteria.

Challenges. Selecting k MCQs from a large pool is difficult because the system must incrementally choose questions that best meet target topic coverage and difficulty. As demonstrated in recent work [20], this sequential decision process naturally suits RL. However, how different RL methods handle varying data distributions and teacher goals remains unclear. Key challenges include designing the MDP—especially rewards that reflect teacher objectives and actions that mimic quiz-design operations—and determining how to train agents for reusable performance across future quiz-design tasks.

Contributions. We define QuizComp, a constrained bi-objective optimization problem that takes MCQs and target topic and difficulty distributions and outputs a satisfying quiz. We design an MDP where each state is a quiz, transitions replace it with one that improves topic or difficulty (reflected in the reward), and actions operate via similarity or dissimilarity on each objective. We implement three RL solutions: DQN, SARSA, and A2C/A3C. DQN serves as the benchmark and has prior use in quiz composition. SARSA enables comparison within temporal difference methods and has shown multitask potential. A2C/A3C provide a structurally different RL approach, combining policy and value networks with potential GPU efficiency benefits.

Empirical validation. Our experiments evaluate multiple RL algorithms across varied input samples and topic/difficulty distributions using synthetic data and the real Med and Math datasets. Results confirm that agents can mimic teacher behavior using only MCQ similarity and dissimilarity. All algorithms use all actions, converge quickly, and tend to favor small gains, showing a bias toward topic and difficulty similarity. This leads to local exploration of MCQ neighborhoods, with larger jumps taken only when similar options are exhausted, mirroring human quiz design. On real datasets, all methods find quizzes highly aligned with targets, with DQN performing best.

We further test transferability across target types, uniform and biased, and datasets, finding strong cross-domain and cross-target transfer. Agents trained on one dataset transfer well to the other, and those trained on biased targets transfer effectively other targets.

MCQ1 [Topic Statistics - Difficulty Easy]
A sample of 8 medical institutions in the country, found these monthly expenses for stationery (in euros): 69, 48, 99, 87, 93, 84, 80, and 98. The expenses of the Red Cross hospital, which was not in the sample, were 1.5 standard deviation below the sample mean. What were the expenses of the Red Cross hospital?

a. 56.75
b. 58.40
c. 106.10
d. 107.75

MCQ2 [Topic Linear Algebra - Difficulty Hard]
Solve the linear system

$$\begin{aligned} x + y + z &= 1 \\ 3x - y - z &= 4 \\ x + 5y + 5z &= -1 \end{aligned}$$

a. The system has (x, x, -2x+1) as solution for every $x \in \mathbb{R}$.
b. The system has no solution.
c. The system has (5/4, y, z) as solution for every $y,z \in \mathbb{R}$.
d. The system has (0, 0, 1) as its unique solution.

MCQ3 [Topic Statistics - Difficulty Hard]
For the summary of a sample given below, identify the possible outliers using the interquartile range.
[MIN Q_1 Median Q_3
MAX; 10 17 18 23 78;]

a. The outlier is 30.
b. The outliers are 17, 18 and 23.
c. The outliers are 10 and 78.
d. The outlier is 78.

MCQ4 [Topic Linear Algebra - Difficulty Easy]
Compute the determinant of
A=([1 -3 1 2; -2 1 2 1; 0 0 -1 0; 2 4 2 1])

a. 11
b. -1
c. 35
d. impossible, the determinant is undefined

MCQ5 [Topic Probabilities - Difficulty Easy]
In how many different ways can you arrange 8 guests around a circular table?

a. 720
b. 120
c. 1024
d. 5040

MCQ6 [Topic Probabilities - Difficulty Hard]
A team of 4 people is to be formed from a group of 7 women and 5 men. How many different teams might be formed?

a. 495
b. 11880
c. 24
d. 479001600

Figure 1: MCQs used to generate quizzes.

Our user study with 28 qualified participants resulted in higher satisfaction and lower effort, confirming the need for automating quiz generation.

2 Problem and Solutions

We consider a set of knowledge topics T and a set M of MCQs. We associate to each MCQ $mcq \in M$ a topic $t \in T$ and a categorical value l that reflects its difficulty level. Following common practice, we consider the difficulty levels in the Bloom taxonomy [7, 11, 16]. Our framework accommodates a variable number of difficulty levels, which depend on the dataset. We define a quiz as a subset $Z \subseteq M$ of MCQs of a fixed size k. We associate with each quiz Z, a topic vector of a fixed size, where each entry i is computed as the proportion of MCQs in Z with topic $t_i \in T$ (recall that each MCQ has a single topic). We also associate to Z a difficulty vector of fixed size, where each entry is computed as the proportion of MCQs in Z that are associated with a given difficulty level in the taxonomy.

A teacher wishing to compose a quiz has in mind topics and difficulty levels to cover. Those are expressed as two target vectors: T_C, a distribution of proportions of MCQs in the quiz with desired topics, and T_D, a distribution of proportions of MCQs in the quiz of desired difficulty levels. For example, $< 0, 0, 0, 0, 0, 0.5, 0, 0, 0.5, 0 >$ is a biased topic vector where half the MCQs cover one topic and the other half another, and $< 0.2, 0.2, 0.2, 0.2, 0.2 >$ represents a uniform vector of difficulties.

We define $topicMatch(Z, T_C)$ and $diffMatch(Z, T_D)$, two functions that reflect to what extent a quiz Z reflects the desired distributions. Our formalization is agnostic to how these functions are defined. In our implementation, we use Cosine similarity.

Problem 1 (The QuizComp Problem). *Given a set M of MCQs, two target vectors T_C and T_D, and an integer k, our goal is to compose a quiz $Z \subseteq M$ of k MCQs s.t.:*

$$\begin{aligned} &\texttt{argmax}_{Z \subseteq M}\ topicMatch(Z, T_C) \\ &\texttt{argmax}_{Z \subseteq M}\ diffMatch(Z, T_D) \end{aligned} \tag{1}$$

2.1 Markov Decision Process Formalization

We assume a Discrete Markov Decision Process (MDP) defined by a triplet $\{\mathcal{S}, \mathcal{A}, \mathcal{R}\}$: State space $\mathcal{S}$ is a set of states of the environment; Action space $\mathcal{A}$ is a set of actions from which the agent selects an action at each step; A reward function $\mathcal{R}$ that computes the reward of an action a_i from state s_i to s'_i, $R_i = r(s_i, a_i, s'_i)$.

States and actions. We define an exploratory agent's environment as a set of distinct quizzes, each containing k MCQs. Although our model is not restricted to pre-existing quizzes, in our implementation, we materialize the space of all possible quizzes to achieve efficiency. The state space represents a quiz Z as a set of k MCQs, and each state is the concatenation of its quiz-level topic and difficulty distribution vectors. When an agent visits a state s (i.e., a quiz Z), it seeks a better state s' (a quiz Z') by applying one of four actions: *SimTopic, SimLevel, DissTopic,* or *DissLevel.* Each action transforms a quiz Z into a new quiz Z' that is either similar or different with respect to topics or difficulty. Section 3.1 details how each action is efficiently implemented.

Reward design. As QuizComp is a multi-objective problem, we propose to define our reward using scalarization, a common approach that transforms the problem into a single objective via a weighted linear sum. Given a quiz Z, we can compute how close it is to the target coverage and difficulty:

$$targetMatch(Z, T_C, T_D) = \alpha \cdot topicMatch(Z, T_C) + (1 - \alpha) \cdot diffMatch(Z, T_D)$$

,
where $\alpha \in [0, 1]$.
We define the reward of taking an action a at a state s:

$$\mathcal{R} \leftarrow targetMatch(s'.Z, T_C, T_D) - targetMatch(s.Z, T_C, T_D)$$

.

A state $s \in \mathcal{S}$ contains a quiz Z, and applying action (a) yields a new state (s') with quiz Z'. The reward captures the agent's progression: if $s'.Z$ is farther from the target than $s.Z$, the reward is negative, penalizing the action and discouraging its future use in similar situations. If $s'.Z$ is closer to the target, the agent receives a positive reward and is encouraged to reuse action a.

Exploration session. An agent learns to navigate in the environment. In each step i, a new quiz Z_{i+1} is composed based on the previous one Z_i by taking an action a_i. An exploration session S, starting at state s_1 (i.e., defines a quiz Z_1), of length n, is a sequence of exploration states and actions: $S = [(s_1, a_1), \ldots, (s_n, a_n)]$.

Reinforcement Learning. Model-free RL [21] addresses sequential optimization by having an agent interact with an environment and maximize cumulative reward. We use this framework for QuizComp, where the agent composes the best quiz (Z) (the best state (s)) by maximizing search progression. RL includes four elements: policy, reward, value function, and environment.

A policy maps perceived states (quizzes) to actions, sometimes via simple functions and sometimes via search, as in QuizComp. The reward function defines the task objective. Maximizing total reward drives policy updates. While rewards capture immediate benefit, the value function reflects long-term desirability. Here, the agent starts from a random quiz Z_1 and moves closer to target topics and difficulties T_C and T_D at each step. Although RL can include a model predicting next states and rewards, we focus on model-free methods.

Policy π. A policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ of an RL agent maps the probability of taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, that is, $\pi(s, a) = Pr(a_t = a | s_t = s)$.

We can rewrite the definition of a session as $S^\pi = [(s_1, \pi(s_1)), \ldots, (s_n, \pi(s_n))]$. By replacing each state by its respective quiz, we rewrite the session S as $S^\pi = [(Z_1, \pi(Z_1)), \ldots, (Z_n, \pi(Z_n))]$.

Optimal Policy π^*. A policy π^* is optimal if its expected cumulative reward is greater than or equal to the expected cumulative reward of all other policies π. The optimal policy has an associated optimal state-value function and optimal Q-function: $Q^*(s, a) \leftarrow max_\pi Q_\pi(s, a)$.

Problem 2 (Revisited QuizComp Problem). *Given a session S, we define sessionMatch(.) to measure the agent's progress toward finding a quiz that matches the target distributions, discounted by $\gamma \in [0, 1]$:*

$$sessionMatch(S, T_C, T_D) = \tag{2}$$

$$\sum_{(s_i, a_i) \in S} \gamma^i \big[targetMatch(s_{i+1}.Z, T_C, T_D) - \tag{3}$$

$$targetMatch(s_i.Z, T_C, T_D) \big] \tag{4}$$

Hence, the problem is to find an optimal policy

$$\pi^* = argmax_\pi \; sessionMatch(S^\pi, T_C, T_D).$$

2.2 Reinforcement Learning Solutions

We explore three RL solutions to solve QuizComp.

Deep Q-Network (DQN) [21] extends Q-learning using deep networks to approximate Q-values, estimating expected cumulative reward for each $\langle s, a \rangle$ pair. It iteratively updates Q-values to balance exploration and exploitation with learning rate α and discount factor γ:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma \max_{a' \in A} Q(s', a') - Q(s, a) \right]. \tag{5}$$

We adopt PER [19] to enrich training experience.

SARSA [21] is an on-policy variant of Q-learning, SARSA updates Q-values using the action the agent actually takes:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma Q(s', a') - Q(s, a) \right]. \tag{6}$$

A2C/A3C [13] are actor–critic methods that combine policy gradients with value estimation. The actor updates action probabilities while the critic estimates advantages:

$$Advantage(s, a) \approx R(s, a, s') + \gamma V(s') - V(s). \tag{7}$$

A3C extends A2C with parallel asynchronous workers, each interacting with its own environment copy to stabilize learning and update shared actor–critic networks.

3 Experiments

4 Related Work

Standard quiz generation. Early rule-based quiz generation used templates and simple rules, such as "include two geometry questions," offering automation but limited generalizability [1, 6]. Later, NLP and ML methods predicted question difficulty [4] and used clustering or topic modeling to group or diversify questions [9]. *These methods focus on single-objective optimization and do not support balancing multiple pedagogical criteria.*

Multi-objective quiz composition. With a high-quality MCQ bank, focus shifts from generation to composition. MOEPG [20] frames exam generation as multi-objective RL. *In our work, we target varied topic and difficulty distributions allowing us to train agents that capture different pedagogical goals. Additionally, we evaluate multiple RL algorithms.*

LLMs for quiz generation. LLMs like GPT-4 generate MCQs text [11, 12], using prompt engineering or chain-of-thought [22], and can generate and evaluate quizzes [12, 15], though irrelevant content may appear. Knowledge Tracing and RAG adapt quizzes to learners [10], while concept-based methods improve grounding [5]. *We use LLMs to generate MCQs but rely on RL for quiz composition to mimic a teacher balancing multiple objectives across data distributions.*

[2]https://en.wikipedia.org/wiki/NASA-TLX

5 Conclusion

We investigated RL-based approaches for quiz composition and provided an extensive performance comparison of DQN, SARSA, and A2C/A3C. Our work investigated and demonstrated effectiveness of transfer learning across datasets and pedagogical targets. Our future work will broaden the action space and study trade-offs between on-the-fly MCQ generation with language models and meeting teacher objectives. We will compare the cost of prompt engineering and model calls to the cost of training an RL agent, including in transfer-learning settings, formalizing boundaries between training and reuse in line with recent work on ML reusability [17].

Impact Statement

This paper advances Online RL by studying its application to teacher-facing services. Our work has notable societal implications, particularly in empowering teachers to understand their materials and generate personalized tests.

References

[1] Tahani Alsubait, Bijan Parsia, and Ulrike Sattler. Ontology-based multiple choice question generation. *KI - Künstliche Intelligenz*, 30(2):183–188, 2016.

[2] Beatriz Flamia Azevedo, Florbela P. Fernandes, Maria F. Pacheco, and Ana I. Pereira. Dataset for Assessing Mathematics Learning in Higher Education, 2024.

[3] Neil De La Fuente and Daniel A Vidal Guerra. A comparative study of deep reinforcement learning models: Dqn vs ppo vs a2c. *arXiv preprint arXiv:2407.14151*, 2024.

[4] W. Feng, P. Tran, S. Sireci, and A. Lan. Reasoning and sampling-augmented mcq difficulty prediction via llms. *arXiv preprint*, arXiv:2503.08551, 2025.

[5] Yicheng Fu, Zikui Wang, Liuxin Yang, Meiqing Huo, and Zhongdongming Dai. Conquer: A framework for concept-based quiz generation, 2025.

[6] Michael Heilman and Noah A. Smith. Good question! statistical ranking for question generation. In *Proceedings of the NAACL-HLT*, 2010.

[7] Kevin Hwang, Sai Challagundla, Maryan M. Alomair, and Lujie Karen Chen. Towards ai-assisted multiple choice question generation and quality evaluation at scale: Aligning with bloom's taxonomy. In *Generative AI for Education (GAIED): Advances, Opportunities, and Challenges. NeurIPS2023*, 2023.

[8] Gonzalo Aguilar Jiménez, Arturo de la Escalera Hueso, and Maria J Gómez-Silva. Reinforcement learning algorithms for autonomous mission accomplishment by unmanned aerial vehicles: A comparative view with dqn, sarsa and a2c. *Sensors*, 23(21):9013, 2023.

[9] Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30(1):121–204, 2020. ERIC: EJ1247652.

[10] Zhaoxing Li, Vahid Yazdanpanah, Jindi Wang, Wen Gu, Lei Shi, Alexandra I. Cristea, Sarah Kiden, and Sebastian Stein. Tutorllm: Customizing learning recommendations with knowledge tracing and retrieval-augmented generation, 2025.

[11] Subhankar Maity, Aniket Deroy, and Sudeshna Sarkar. Exploring the capabilities of prompted large language models in educational and assessment applications, 2024.

[12] N. Meißner, S. Speth, J. Kieslinger, and S. Becker. Evalquiz — llm-based automated generation of self-assessment quizzes in se education. In *SEUH 2024*, 2024.

[13] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016.

[14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[15] Sérgio Silva Mucciaccia, Thiago Meireles Paixão, Filipe Wall Mutz, Claudine Santos Badue, Alberto Ferreira de Souza, and Thiago Oliveira-Santos. Automatic multiple-choice question generation and evaluation systems based on LLM: A study case with university resolutions. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2246–2260, Abu Dhabi, UAE, January 2025.

[16] Deepak Subramani Nicy Scaria, Suma Dharani Chenna. Automated educational question generation at different bloom's skill levels using large language models: Strategies and evaluation. In *Artificial Intelligence in Education. AIED 2024*, 2024.

[17] Sepideh Nikookar, Sohrab Namazi Nia, Senjuti Basu Roy, Sihem Amer-Yahia, and Behrooz Omidvar-Tehrani. Model reusability in reinforcement learning. *VLDB J.*, 34(2):283–304, 2025.

[18] Andrew M. Olney. Generating multiple choice questions from a textbook: Llms match human performance on most metrics. In *Proceedings of the Workshop on Empowering Education with LLMs, co-located with AIED 2023, Tokyo, Japan, July 7, 2023, volume 3487, pages 111–128.*, 2023.

[19] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, Puerto Rico, May 2-4, 2016*, 2016.

[20] Y. Shang, X. Luo, L. Wang, H. Peng, X. Zhang, Y. Ren, and K. Liang. Reinforcement learning guided multi-objective exam paper generation. *arXiv preprint*, arXiv:2303.01042, 2023.

[21] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, 2018.

[22] Z. Yao, C. Zhang, J. Gu, J. Yin, Z. Yin, and M. Tan. Mcqg-srefine: Multiple choice question generation with iterative self-critique and correction. *arXiv preprint*, arXiv:2410.13191, 2024.