From Questions to Assessment Tuples: A Multi-Agent Framework with Bloom-Specialized Agents and Automated Verification

Gee-Lyle Wong **Runcong Zhao** **Yulan He** **Jiazheng Li**
King's College London
{gee-lyle.wong, runcong.zhao, yulan.he, jiazheng.li}@kcl.ac.uk

Abstract

Automatic question generation with large language models has advanced rapidly, yet producing *assessment-ready* items, complete with mark schemes and expected answers, remains challenging, especially when generation must reliably target higher-order cognitive levels in Bloom's Taxonomy. We propose a multi-agent, multi-stage framework that generates structured assessment tuples for both short-answer questions (SAQs) and scenario-based questions (SBQs), combining Bloom-specialized generation agents with staged decomposition and automated verification. We further introduce a rubric-guided LLM-as-a-judge evaluation framework with Bloom-specific alignment metrics. Experiments on university-level AI course material across five generation pipelines show that prompt-level Bloom conditioning alone is insufficient to reliably achieve cognitive control. In contrast, our structured approach yields consistent and notable improvements in alignment, mark scheme quality, and output yield, particularly for higher-order Bloom levels over baseline pipelines.

1 Introduction

Assessment is central to teaching and learning, supporting formative practice, diagnostic feedback, and summative evaluation. However, producing high-quality assessment items is time-intensive and requires both domain expertise and pedagogical design, limiting scalability (Kurdi et al., 2020; Das et al., 2021). Automatic Question Generation (AQG) and Automatic Item Generation (AIG) have therefore emerged as key research directions, evolving from rule-based systems to neural and, more recently, large language model (LLM)-based approaches (Zhang et al., 2021; Kurdi et al., 2020; Tan et al., 2025).

Despite this progress, most work focuses on generating question text alone or on multiple-choice questions (MCQs), leaving open-ended assessment types such as short-answer questions (SAQs) and scenario-based questions (SBQs) comparatively underexplored (Song et al., 2025; Chan et al., 2025). SAQs require brief, constructed responses that demonstrate conceptual understanding, while SBQs present realistic, course-grounded scenarios that require students to apply their knowledge and reasoning using both the provided context and learned material.

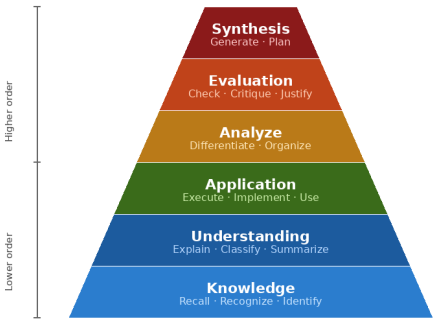

Figure 1: Bloom's Taxonomy of Educational Objectives (LW et al., 2001), comprising six cognitive levels ordered by increasing complexity. Reliably generating assessment items at higher-order levels is a central challenge addressed in this work.

Open-ended formats play a critical role in assessing deeper understanding and reasoning, beyond the constrained nature of MCQs (Kurdi et al., 2020; Das et al., 2021; Song et al., 2025). However, their effectiveness depends on more than the question alone. Reliable assessment requires aligned marking schemes and exemplar answers, introducing additional complexity: systems must generate coherent, pedagogically grounded assessment items with clearly scoped tasks, objective grading criteria, and internal consistency across components. This shifts the problem from question generation to *structured assessment generation*, where multiple components must jointly satisfy pedagogical constraints.

A second challenge lies in *cognitive control*. Bloom's Taxonomy is widely used to define learning objectives, and prior work has explored Bloom-

Proceedings of the 21st Workshop on Innovative Use of NLP for Building Educational Applications, pages 292–335

aware prompting to control question difficulty. While such approaches improve over unconstrained generation, they remain unreliable, particularly at higher-order cognitive levels such as *analyze*, *evaluate*, and *synthesis* (Scaria et al., 2024; Maity et al., 2024; Yaacoub et al., 2025). More broadly, aligning generated items with intended cognitive demand remains an open problem (Uto et al., 2023).

A third limitation concerns *evaluation*. Question generation is inherently one-to-many, making reference-based metrics, such as lexical overlap, unreliable proxies for quality (Oh et al., 2023; Nguyen et al., 2024). Existing evaluation approaches, ranging from reference-free metrics to LLM-as-a-judge, remain fragmented and fail to capture whether generated items are coherent, gradable, and cognitively aligned as complete assessment units (Mohammadshahi et al., 2023; Liu et al., 2023; Hashemi et al., 2024).

To address these challenges, we propose a unified, Bloom-controlled framework for generating assessment-ready items from course material. Generation is decomposed into a structured multi-stage process that produces complete assessment tuples: for SAQs, a question, mark scheme, and expected answer; for SBQs, an additional scenario is constructed and optionally refined. Bloom-specialized generation enables explicit cognitive control, while automated verification enforces grounding, alignment, and internal consistency. We further introduce a rubric-guided LLM-as-a-judge evaluation framework tailored to structured assessment generation. The framework separates stem-level quality from full assessment consistency, enabling fine-grained analysis of failure modes. We also define Bloom-specific metrics that quantify alignment across all cognitive levels and within higher-order levels.

We evaluate five pipelines spanning zero-shot, Bloom-aware, and multi-stage approaches across both SAQs and SBQs using Artificial Intelligence course material. Results show that prompt-level Bloom conditioning alone does not achieve consistent cognitive targeting in our setting, while staged generation with Bloom specialization and verification yields improvements in alignment, structural quality, and output yield, particularly at higher-order levels.

Our contributions are as follows:

- We introduce a Bloom-specialized, multi-agent, multi-stage framework for both SAQs and SBQs, including scenario construction and refinement for contextualized reasoning.
- We develop a multi-criteria LLM-as-a-judge framework that evaluates pedagogical quality, structural validity, and Bloom alignment.
- We conduct an empirical comparison of five generation pipelines across two assessment formats, showing that staged Bloom specialization with verification improves alignment, structural quality, and yield over prompt-level conditioning.

2 Related Work

Automatic question and item generation. Recent surveys highlight the flexibility of LLMs for generating assessment items across domains, but also emphasize limitations in quality control, pedagogical grounding, and evaluation practices (Kurdi et al., 2020; Zhang et al., 2021; Das et al., 2021; Tan et al., 2025; Song et al., 2025). Much of this work focuses on multiple-choice questions, while open-ended formats such as SAQs and SBQs remain comparatively underexplored despite their importance for assessing higher-order understanding (Song et al., 2025).

Assessment consistency and explainable scoring. Related work on automated student answer scoring and feedback generation shows the importance of aligning outputs with rubrics, key answer elements, rationales, and curriculum context. Prior systems use ChatGPT-generated rationales, thought-tree decomposition, preference calibration, and educator-in-the-loop workflows to improve explainability and assessment reliability (Li et al., 2023, 2024b,a; Zhao et al., 2025). These works motivate our focus on generating assessment-ready tuples rather than question stems alone.

Cognitive control and Bloom's taxonomy. Controlling cognitive complexity is a central goal in educational generation. Bloom-aware prompting can improve alignment compared to standard prompting, but performance remains inconsistent, particularly for higher-order categories (Scaria et al., 2024; Maity et al., 2024; Yaacoub et al., 2025; Uto et al., 2023; Yadav et al., 2025). Our work therefore treats Bloom control as part of the generation pipeline, rather than relying solely on prompt-level conditioning.

Evaluation of question generation. Evaluation remains challenging because question generation is inherently one-to-many. Metrics that compare generated questions against fixed reference questions, such as BLEU- or ROUGE-style overlap metrics, correlate poorly with human judgment and may penalize valid alternative questions, motivating reference-free, multi-dimensional, and LLM-

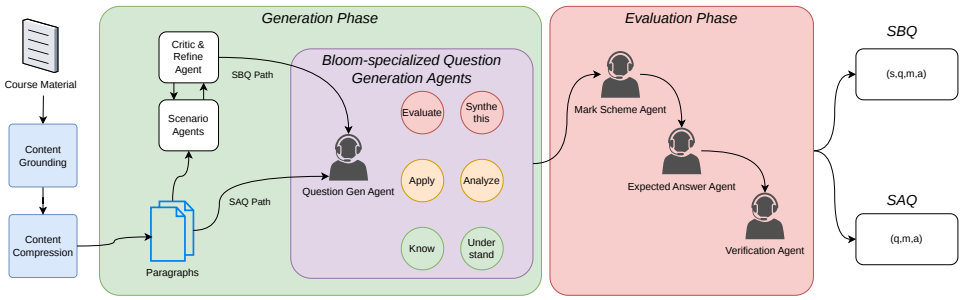

Figure 2: Overview of the proposed Bloom-guided multi-agent generation framework. The pipeline begins with course grounding and relevance filtering, followed by course-content compression. For SBQs, a dedicated scenario generation stage is applied and refined through critique before question generation; for SAQs, question generation proceeds directly from the filtered and compressed course content. Both paths then generate a mark scheme and expected answer, after which a verification agent checks Bloom alignment, grounding, and structural validity. Final outputs are tuples (q, m, a) for SAQs and (s, q, m, a) for SBQs.

based alternatives (Oh et al., 2023; Nguyen et al., 2024; Mohammadshahi et al., 2023; Fu et al., 2024; Liu et al., 2023; Hashemi et al., 2024). However, some methods often focus on question-level quality rather than full assessment consistency, so our evaluation combines stem- and scenario-level criteria with triple-level checks across the question, mark scheme, and expected answer, alongside Bloom-specific alignment metrics.

Structured and multi-stage generation. Prior work has decomposed educational assessment into multiple components, including question, answer, and distractor generation in MCQ pipelines, or key-element matching, rubric application, rationale generation, and refinement in explainable scoring systems (Li et al., 2024b, 2025). Iterative refinement and self-feedback further motivate the use of critique and verifier models for improving generated outputs (Madaan et al., 2023; Li et al., 2025). Our work extends this structured view to open-ended assessment generation by constructing full assessment tuples through multi-stage generation with explicit verification.

Limitations of existing approaches. Across these directions, three limitations persist: (i) *limited support for assessment-ready outputs beyond question text*, (ii) *unreliable control over cognitive complexity, particularly at higher-order levels*, and (iii) *fragmented evaluation practices that do not capture full assessment quality*. Our work addresses these gaps through a unified framework that integrates structured generation, Bloom-specialized control, and rubric-based evaluation for complete assessment items.

3 Methodology

We propose a unified, Bloom-controlled generation framework for producing assessment-ready items from course documents. Rather than generating questions in isolation, the framework produces structured assessment tuples, enabling both cognitive control during generation and structural validation after generation. It supports two assessment variants within the same system: short-answer questions (SAQs), generated directly from source content, and scenario-based questions (SBQs), which introduce an intermediate scenario representation prior to question generation.

Formally, let D denote an extracted course document, and let $b \in \mathcal{B}$ denote a target Bloom level (knowledge, comprehension, application, analysis, evaluation, synthesis)[1]. The framework generates an assessment output y, defined as $y = (q, m, a)$ for SAQs and $y = (s, q, m, a)$ for SBQs, where s is a scenario, q is the question, m is the mark scheme, and a is an expected full-mark answer.

The overall system can be viewed as a controlled generation function

$$\mathcal{F}(D, b, t) \rightarrow y,$$

where $t \in \{\texttt{SAQ}, \texttt{SBQ}\}$ specifies the assessment type.

3.1 Generation Framework

The framework is formulated as a multi-agent, multi-stage generation process. Rather than relying on a single prompt to jointly produce all outputs, generation is decomposed into specialized components that each operate on a narrower subtask. This decomposition serves three purposes: it improves control over Bloom level, reduces interference between subtasks, and enables explicit verification of intermediate and final outputs.

[1]We follow the original Bloom's Taxonomy (LW et al., 2001) with six cognitive levels ordered by increasing complexity.

At a high level, the framework consists of four functional stages:

1. **Content grounding:** identify the source content from which an item should be generated.
2. **Question construction:** generate either a direct question (SAQ) or a scenario followed by a question (SBQ), conditioned on a target Bloom level.
3. **Assessment completion:** generate a mark scheme and then an expected answer conditioned on the previously generated outputs.
4. **Verification and filtering:** apply automated checks to enforce Bloom alignment, grounding, and internal consistency.

This design treats SAQs and SBQs as two paths through the same framework. Both share the same downstream structure (question $\rightarrow$ mark scheme $\rightarrow$ expected answer), while SBQs introduce an additional scenario construction and refinement step before question generation.

3.2 Framework Components

Bloom-specialized question agents. To improve cognitive control, the framework uses Bloom-specialized generation instructions. For each target level b, a dedicated question-generation agent is configured with level-specific guidance about the expected cognitive operation, response format, and acceptable task type. This replaces a single generic prompting strategy with a family of specialized generators $G_q^{(b)}(\cdot)$ for $b \in \mathcal{B}$.

For SAQs, questions are generated directly from source content as $q = G_q^{(b)}(D)$, while for SBQs, question generation is conditioned on a refined scenario, $q = G_q^{(b)}(s')$. This per-level specialization is central to the framework, as the objective is not only to generate valid questions, but to control their cognitive demand.

Scenario construction agent (SBQ variant). SBQs introduce an intermediate contextual representation. A scenario agent generates a course-grounded situation $s = G_s(D)$, intended to provide a context in which subsequent reasoning is situated. Rather than adding superficial realism, this stage transforms source knowledge into a setting that supports application-, analysis-, or evaluation-oriented assessment.

Because unconstrained scenario generation may introduce answer leakage or implicit guidance, the generated scenario is not used directly and is instead passed to a refinement stage.

Scenario critique–refinement agent (SBQ variant). The framework incorporates a constraint-aware critique–refinement stage. A critique function $V_s(\cdot)$ evaluates the generated scenario with respect to the source content, producing a validation signal $r_s = V_s(D, s)$ that reflects compliance with pedagogical constraints such as neutrality, absence of answer leakage, and suitability for downstream questioning. If violations are detected, the scenario is revised as $s' = R_s(D, s, r_s)$; otherwise, $s' = s$. This ensures that the scenario remains a neutral, information-bearing context rather than implicitly encoding the answer.

Mark scheme generation agent. Given a generated question q and context c (where $c = D$ for SAQs and $c = s'$ for SBQs), the framework generates a structured mark scheme $m = G_m(c, q)$. The mark scheme is represented as a set of atomic marking points with associated integer marks, enabling transparent grading and facilitating downstream verification.

Expected answer generation agent. The final generation stage produces an exemplary full-mark answer $a = G_a(c, q, m)$ conditioned on the context, question, and mark scheme. Conditioning on the mark scheme ensures that the generated answer aligns with the grading criteria rather than being only superficially plausible.

Verification agent. All candidate outputs are passed through a final verification layer before being retained. The verifier evaluates whether generated components satisfy structural and pedagogical constraints, formalized as

$$\phi(y; D, b, t) \in \{0, 1\}.$$

Only outputs with $\phi = 1$ are retained. This stage functions as a quality-control mechanism and is a core component of the framework rather than a post-hoc filtering step.

3.3 SAQ and SBQ Instantiations

Although the framework is unified, the two assessment formats differ in how the question context is constructed.

SAQ instantiation. For SAQs, question generation is performed directly from course content. Formally, we define $q = G_q^{(b)}(D)$, $m = G_m(D, q)$, and $a = G_a(D, q, m)$. The candidate tuple $y = (q, m, a)$ is then passed through verification:

$$\mathcal{F}_{\text{SAQ}}(D, b) = y \cdot \mathbb{I}[\phi(y; D, b) = 1],$$

where only tuples satisfying the verification predicate ϕ are retained as valid outputs. This variant is

suited to direct recall, explanation, application, or short analytical reasoning grounded in the provided course material.

SBQ instantiation. For SBQs, the framework first constructs and conditionally refines a scenario before question generation. Specifically, we define $s = G_s(D)$ and $r_s = V_s(D, s)$, where $V_s(\cdot)$ evaluates the scenario against pedagogical constraints. The refined scenario is then given by

$$s' = \begin{cases} R_s(D, s, r_s) & \text{if } r_s \text{ indicates a violation,} \\ s & \text{otherwise.} \end{cases}$$

Question generation then proceeds as $q = G_q^{(b)}(s')$, followed by $m = G_m(s', q)$ and $a = G_a(s', q, m)$. The candidate tuple $y = (s', q, m, a)$ is then verified:

$$\mathcal{F}_{\text{SBQ}}(D, b) = y \cdot \mathbb{I}[\phi(y; D, b) = 1].$$

The scenario introduces a context in which students must interpret information before answering, enabling contextualized application and higher-order reasoning while preserving the same downstream assessment structure as SAQs.

3.4 Control and Verification Logic

The framework combines *generation control* and *verification*. Generation control is achieved through Bloom-specialized agents and staged conditioning, while verification acts as a filtering mechanism that enforces structural and pedagogical validity of the generated outputs.

Given a candidate output y, the verification stage evaluates whether it satisfies a set of core constraints grounded in the source content and target Bloom level. These checks cover four aspects: (i) **Bloom alignment**, ensuring the question matches the intended cognitive level; (ii) **grounding**, requiring all generated components to be supported by their source context (D for SAQs, and both D and s' for SBQs); (iii) **mark scheme quality**, enforcing completeness, non-redundancy, and alignment with the question; and (iv) **answer fidelity**, ensuring the expected answer satisfies all marking criteria without introducing unsupported information.

Only candidates that satisfy all constraints are retained. The framework therefore operates not as a single generator, but as a controlled generation system in which outputs are iteratively produced and selectively accepted based on explicit validation criteria.

3.5 Discussion of Design Rationale

The framework is designed around the observation that assessment generation is inherently compositional. A high-quality item is not only a well-phrased question; it is a coherent structure linking source content, cognitive demand, grading criteria, and an exemplary answer. Treating these components as jointly generated in one step makes control difficult and validation opaque. By decomposing the task into specialised agents with explicit interfaces, the proposed framework improves controllability, makes failures more localisable, and supports direct comparison between simpler and more structured generation strategies in the experimental section. Full prompt templates, per-stage implementation details, Bloom-level generation guidelines, and the generation-time verification predicate are provided in Appendix A.

4 Experiments and Results

4.1 Experiment Setup

We evaluate automatic question generation on university level *Artificial Intelligence* course materials derived from publicly available sources. The source documents are preprocessed into structured content representations, from which assessment items are generated.

Experiments are conducted across two assessment formats: short-answer questions (SAQs) and scenario-based questions (SBQs). While both formats aim to assess conceptual understanding, SBQs additionally require the generation of a contextual scenario that conditions the question, enabling applied and higher-order reasoning.

Generation is performed using **GPT-4.1**, **GPT-4o-mini**, and **Gemini-2.5 Flash** (OpenAI et al., 2024; Comanici et al., 2025). Results are reported for each pipeline-generator pair (p, g) to isolate the effect of both prompting strategy and underlying model capability. Evaluation is performed using three LLM-as-a-judge models: **GPT-4o-mini**, **GPT-4.1-mini**, and **Gemini-2.5 Flash**. Unless otherwise stated, reported criterion and Bloom scores are averaged across these evaluator models.

4.2 Baselines and Proposed Pipelines

We evaluate five generation pipelines designed to progressively address limitations in controllability, decomposition, and cognitive alignment.

Zero-shot (ZS). A single prompt generates the full assessment item in one step. For SAQs, this corresponds to generating a (question, mark scheme,

expected answer) triple, while for SBQs the scenario is generated jointly with the question.

Zero-shot + Bloom (ZS+B). Extends zero-shot prompting by explicitly conditioning on a target Bloom level, testing whether cognitive alignment can be achieved through prompt-level control alone.

Multi-stage zero-shot (MZS). Decomposes generation into sequential stages (question → mark scheme → answer), reducing task interference. For SBQs, scenario and question generation are performed jointly in the first stage.

Multi-stage zero-shot + Bloom (MZS+B). Integrates Bloom-level conditioning into the multi-stage pipeline during question generation, assessing whether decomposition and cognitive control jointly improve alignment.

Proposed pipeline. Our approach extends the multi-stage Bloom-aware pipeline in three ways. First, question generation uses *Bloom-specialized agents* with per-level system instructions rather than a single generic Bloom prompt. Second, for SBQs, a dedicated scenario generation stage is followed by critique and refinement to enforce grounding, neutrality, and the absence of answer leakage. Third, an explicit verification layer filters candidates based on Bloom alignment, grounding, and internal consistency.

4.3 Evaluation Criteria and Metrics

We define a structured set of criteria capturing both *question quality* and *assessment consistency*, evaluated at two levels: the question stem (or scenario and question for SBQs) and the full assessment triple. All criteria are evaluated independently using rubric-guided LLM judges to reduce cross-criterion interference. Scores are aggregated across three evaluator models to test whether the main conclusions are robust across different evaluators. We use the following abbreviations throughout the paper.

Stem-level criteria. **CG** (course grounding): whether the question is derived from source material. **Cla.** (clarity): linguistic precision and absence of ambiguity. **STI** (single-task integrity): coherence of the assessed objective. **OG-S** (objective gradability, stem): whether the question admits a clearly gradable answer. For SBQs, we additionally evaluate **SRN** (scenario relevance/necessity) and **SG** (scenario grounding).

Triple-level criteria. **OG-T** (objective gradability, triple-aware): whether the full triple supports consistent and objective grading. **MSQ** (mark scheme quality): coverage, structure, and non-redundancy of marking points. **AF** (answer fidelity): alignment between the expected answer and the mark scheme.

Bloom-specific metrics. **BLA** (Bloom level alignment): a soft alignment score accounting for the hierarchical structure of Bloom's Taxonomy, defined as $\mathrm{BLA}_i = 1 - |\mathrm{gap}(b_i, \hat{b}_i)|/L$, where b_i is the target level, $\hat{b}_i$ the inferred level, and L the maximum possible gap. **BA** (Bloom accuracy): exact-match alignment between intended and inferred Bloom levels. **HOSR** (higher-order success rate): exact-match alignment restricted to higher-order levels ($\{\texttt{analysis}, \texttt{evaluation}, \texttt{synthesis}\}$). Full formal definitions are provided in Appendix B.

4.4 Evaluation Protocol

We employ three complementary evaluation paradigms: criterion-based scoring, matched-task pairwise comparison, and inter-judge agreement analysis.

Criterion-based evaluation. Each generated item receives a discrete score $s \in \{0, 1, 2\}$ per criterion from each evaluator model, normalized to $[0, 1]$ and averaged across judges. We aggregate using discrimination-weighted scoring, where criteria that better separate pipeline performance contribute proportionally more to the final composite score. Full formalization is given in Appendix B.

Matched-task pairwise comparison. To compare pipelines directly, we match them on task slots defined by source file, target Bloom level, and generation model. For each matched triple, we compare representative outputs in two modes: *raw* (first generated candidate, measuring intrinsic quality) and *production* (highest-scoring candidate, measuring end-to-end pipeline quality after selection). Outcomes are aggregated into win, tie, and loss rates with bootstrap 95% confidence intervals.

Inter-judge agreement. To examine whether the evaluation is stable across LLM judges, we compute agreement statistics over GPT-4o-mini, GPT-4.1-mini, and Gemini-2.5 Flash. For rubric criteria, we report exact three-judge agreement, the mean standard deviation of normalized scores, and mean pairwise Pearson correlation. For inferred Bloom level, we additionally report exact three-judge agreement and mean pairwise agreement over predicted Bloom labels.

4.5 Results

Overview Tables 1–5 summarize criterion-level performance, Bloom metrics, inter-judge agreement, and matched-task pairwise comparisons

		SAQ							SBQ									
Pipe.	**Generator**	**OG-T**	**OG-S**	**MSQ**	**AF**	**STI**	**Cla.**	**CG**	**OG-T**	**OG-S**	**SRN**	**SG**	**MSQ**	**AF**	**STI**	**Cla.**	**CG**	**Avg**
ZS	GPT-4.1	0.931	0.896	0.519	0.959	0.918	0.991	0.981	0.927	0.827	0.555	0.845	0.515	0.976	0.858	0.903	0.924	0.845
	GPT-4o-mini	0.959	0.921	0.447	0.959	0.921	0.991	0.981	0.852	0.826	0.595	0.894	0.549	0.962	0.845	0.909	0.955	0.848
	Gemini-2.5 Flash	0.961	0.936	0.426	0.980	0.917	0.995	0.946	0.919	0.820	0.599	0.829	0.653	0.964	0.833	0.887	**0.964**	0.852
ZS+B	GPT-4.1	0.950	0.925	0.431	0.965	0.918	0.981	0.959	0.904	0.825	0.610	0.842	0.523	0.980	0.879	0.873	0.929	0.843
	GPT-4o-mini	0.943	0.912	0.236	0.953	0.918	0.991	0.975	**0.946**	0.833	0.588	0.858	0.603	0.956	0.912	**0.956**	0.931	0.844
	Gemini-2.5 Flash	**0.977**	0.921	0.398	0.944	0.921	**1.000**	0.986	0.880	**0.870**	**0.681**	0.787	0.662	0.968	0.856	0.856	0.954	0.854
MZS	GPT-4.1	0.863	0.877	0.352	0.944	0.896	0.986	0.957	0.888	0.818	0.551	0.827	0.554	**0.990**	0.856	0.876	0.939	0.823
	GPT-4o-mini	0.867	0.892	0.296	0.963	0.885	0.981	0.975	0.875	0.808	0.609	0.863	0.553	0.979	0.854	0.896	0.931	0.827
	Gemini-2.5 Flash	0.890	0.912	0.269	0.912	0.922	0.996	0.971	0.881	0.804	0.619	0.769	0.648	0.961	0.929	0.854	0.936	0.830
MZS+B	GPT-4.1	0.840	0.908	0.377	0.944	0.931	0.994	0.975	0.880	0.822	0.597	0.853	0.624	**0.990**	0.899	0.897	0.930	0.841
	GPT-4o-mini	0.879	0.920	0.287	0.957	0.906	0.993	0.969	0.926	0.833	0.637	0.902	0.738	0.961	0.890	0.955	0.943	0.856
	Gemini-2.5 Flash	0.871	0.947	0.338	0.914	0.922	0.996	0.975	0.866	0.808	0.637	0.785	0.549	0.986	0.910	0.854	0.926	0.830
Ours	GPT-4.1	0.935	0.925	**0.918**	0.989	0.985	0.987	0.983	0.788	0.774	0.603	0.908	0.784	0.884	0.963	0.836	0.849	0.882
	GPT-4o-mini	0.942	0.930	0.903	0.990	0.971	0.983	0.983	0.815	0.814	0.618	**0.940**	**0.814**	0.905	0.970	0.884	0.854	0.895
	Gemini-2.5 Flash	0.975	**0.950**	0.897	**0.994**	**0.989**	0.993	**0.988**	0.860	0.841	0.605	0.885	0.779	0.929	**0.981**	0.888	0.892	**0.903**

Table 1: Criterion-level results across pipelines, generation models, and task types, averaged across the three evaluator models. SAQ criteria: OG-T (objective gradability, triple-aware), OG-S (objective gradability, stem-only), MSQ (mark scheme quality), AF (answer fidelity), STI (single-task integrity), Cla. (clarity), and CG (course grounding). SBQ extends SAQ with two scenario-specific criteria: SRN (scenario relevance/necessity) and SG (scenario grounding). Avg is the macro-average over all 16 non-Bloom criteria. **Bold** = best per column; underline = second.

	SAQs									SBQs									
	GPT-4.1			GPT-4o-mini			Gemini-2.5 Flash			GPT-4.1			GPT-4o-mini			Gemini-2.5 Flash			Avg
Pipe.	BLA	BA	HOSR	BLA	BA	HOSR	BLA	BA	HOSR	BLA	BA	HOSR	BLA	BA	HOSR	BLA	BA	HOSR	
ZS	0.781	0.019	0.000	0.747	0.057	0.000	0.771	0.000	0.000	0.775	0.091	0.111	0.759	0.205	0.333	0.784	0.270	0.000	0.317
ZS+B	0.894	0.736	0.750	0.891	0.679	0.500	0.889	0.694	0.357	0.864	0.576	0.640	0.894	0.676	0.833	0.894	0.694	0.636	0.728
MZS	0.765	0.042	0.045	0.795	0.075	0.158	0.772	0.071	0.000	0.745	0.089	0.074	0.767	0.181	0.167	0.745	0.137	0.000	0.313
MZS+B	0.885	0.637	0.750	0.899	0.681	0.727	0.899	0.722	0.750	0.853	0.558	0.641	0.879	0.625	0.815	0.850	0.514	0.467	0.731
Ours	**0.985**	**0.947**	**0.919**	**0.997**	**0.991**	**0.989**	**0.961**	**0.880**	**0.884**	**0.981**	**0.938**	**0.943**	**0.988**	**0.964**	**0.975**	**0.939**	**0.847**	**0.811**	**0.941**

Table 2: Bloom-related metrics across pipelines, generation models, and assessment formats. Columns are grouped by assessment format and generation model, and rows correspond to pipelines. BLA denotes the soft Bloom alignment score, while BA and HOSR are derived from the aggregated inferred Bloom level for each item. BA denotes exact Bloom alignment accuracy, and HOSR denotes higher-order success rate. Avg is the macro-average over all displayed Bloom metrics across formats and generators. Higher is better for all metrics. **Bold** = best per column; underline = second best.

across all pipeline–generator configurations and both assessment formats. Overall, the proposed pipeline remains the strongest system across the main evaluation settings. Its advantage is clearest for SAQs, where it dominates both criterion-level quality and pairwise win rates. For SBQs, the proposed pipeline is strongest in Bloom control and scenario grounding, but it does not uniformly dominate simpler baselines on all other criteria. In particular, objective gradability and answer fidelity remain competitive across pipelines, suggesting that simpler scenario-based items can sometimes be easier to grade and align more directly with their expected answers.

Assessment Quality The clearest criterion-level separation is mark scheme quality (Table 1). For SAQs, the proposed pipeline achieves MSQ scores of 0.897–0.918 across generators, compared with 0.236–0.519 for the baselines, indicating that the decomposition and verification stages substantially improve the coverage and structure of marking points. The same pattern holds for SBQs, although with a smaller margin: the proposed pipeline reaches 0.779–0.814, while the strongest baseline reaches 0.738.

Other criteria show a more nuanced pattern. For SAQs, the proposed pipeline remains consistently strong on AF (0.989–0.994), STI (0.971–0.989), and CG (0.983–0.988), suggesting that the generated answers, mark schemes, and source grounding remain internally consistent. However, for SBQs, the proposed pipeline does not uniformly dominate simpler baselines on all non-Bloom criteria. Some baselines obtain higher OG-T, OG-S, and AF scores, suggesting that the added contextual complexity of scenarios can make objective gradability and answer alignment harder to satisfy consistently. Overall, the proposed pipeline improves the more structured dimensions of assessment construction, especially MSQ and scenario grounding,

Fmt.	Pipe.	n	Rub. Ag.	BLA Ag.	Std.	r	Lvl Ex./Pair
SAQ	ZS	140	0.773	0.864	0.060	0.314	0.636 / 0.750
	ZS+B	142	0.774	0.725	0.059	0.394	0.718 / 0.807
	MZS	261	0.724	0.782	0.072	**0.436**	0.663 / 0.768
	MZS+B	228	0.747	0.693	0.068	0.384	0.684 / 0.782
	Ours	1072	**0.876**	**0.880**	**0.033**	0.356	**0.878 / 0.916**
SBQ	ZS	136	0.645	0.640	**0.098**	0.445	0.529 / 0.669
	ZS+B	129	0.652	0.597	0.101	0.472	0.597 / 0.716
	MZS	246	0.615	0.622	0.109	0.382	0.524 / 0.667
	MZS+B	214	0.633	0.537	0.107	0.388	0.500 / 0.653
	Ours	750	**0.667**	**0.860**	**0.098**	**0.484**	**0.860 / 0.905**

Table 3: Inter-judge agreement across GPT-4o-mini, GPT-4.1-mini, and Gemini-2.5-Flash by assessment format and pipeline. Rub. Ag. is macro-average exact agreement over non-Bloom rubric criteria. BLA Ag. is exact agreement over the discretized soft Bloom-alignment score. Std. is the macro-average standard deviation of normalized non-Bloom rubric scores; r is mean pairwise Pearson correlation over non-Bloom rubric scores. Lvl Ex./Pair reports exact three-judge agreement / mean pairwise agreement on the inferred Bloom-level label. **Bold** = best per column; underline = second best.

Fmt.	Crit.	Agree	Std.	r
SAQ	OG-T	0.662	0.086	0.250
	OG-S	0.649	0.083	0.266
	MSQ	0.753	0.081	**0.740**
	AF	0.868	0.043	0.314
	STI	0.635	0.086	0.290
	Cla.	**0.954**	**0.011**	0.189
	CG	0.931	0.017	0.587
	BLA	0.789	0.032	0.595
SBQ	OG-T	0.602	0.129	0.388
	OG-S	0.500	0.144	0.365
	SRN	*0.321*	0.166	0.302
	SG	0.650	0.085	0.455
	MSQ	0.711	0.099	**0.723**
	AF	**0.925**	**0.029**	0.310
	STI	0.580	0.101	0.348
	Cla.	0.670	0.124	0.421
	CG	0.823	0.050	0.595
	BLA	0.651	0.047	0.570

Table 4: Criterion-level inter-judge agreement, macro-averaged over pipelines. Agree is exact three-judge agreement; Std. is the mean standard deviation of normalized scores across judges; r is the mean pairwise Pearson correlation. For non-Bloom criteria, values are computed over rubric scores. For BLA, values are computed separately over the discretized soft Bloom-alignment score. SAQ criteria: OG-T, OG-S, MSQ, AF, STI, Cla., CG, and BLA. SBQ additionally includes SRN and SG. **Bold** = best per column within each format; underline = second best.

while SBQ generation remains intrinsically more challenging.

Bloom Cognitive Control The strongest and most consistent gains occur in Bloom-level control (Table 2). Pipelines without explicit Bloom conditioning have very low exact Bloom accuracy across generators: ZS reaches BA values of only 0.000–0.057 for SAQs and 0.091–0.270 for SBQs, while MZS remains similarly low at 0.042–0.075 and 0.089–0.181 respectively. This indicates that decomposition alone does not provide reliable cognitive-level control.

Adding Bloom prompts substantially improves alignment, but remains less reliable than the proposed pipeline. ZS+B and MZS+B improve BA across both formats, but the proposed pipeline achieves the highest Bloom metrics for every generator and assessment format. For SAQs, it reaches BA values of 0.880–0.991 and HOSR values of 0.884–0.989; for SBQs, it reaches BA values of 0.847–0.964 and HOSR values of 0.811–0.975.

The macro-average column further shows that this improvement is not driven by a single generator or format. The proposed pipeline achieves the highest overall Bloom average, 0.941, compared with 0.731 for MZS+B and 0.728 for ZS+B. This suggests that Bloom-specialized generation and verification improve cognitive targeting across model families, especially for higher-order levels where prompt-only conditioning is less consistent.

Scenario Construction For SBQs, scenario grounding is highest under the proposed pipeline across all generators, with SG scores of 0.908, 0.940, and 0.885 for GPT-4.1, GPT-4o-mini, and Gemini-2.5 Flash respectively (Table 1). This supports the value of the dedicated scenario generation, critique, and refinement stages in anchoring scenarios to the source material.

However, SRN remains less clearly improved. The proposed pipeline scores 0.603–0.618, while baselines range from 0.551 to 0.681, suggesting that the pipeline improves scenario grounding more reliably than scenario necessity (Table 1). The agreement results support this interpretation: SRN has the lowest SBQ judge agreement (0.321) and highest standard deviation (0.166), indicating that scenario necessity is both difficult to generate and difficult to judge consistently (Table 4).

Inter-Judge Agreement At the pipeline level, the proposed method achieves the highest rubric agreement, discretized BLA agreement, and inferred Bloom-level agreement for both formats (Table 3). For SAQs, these are 0.876, 0.880, and 0.878/0.916 exact/pairwise agreement respectively; for SBQs, they are 0.667, 0.860, and 0.860/0.905. This strengthens the Bloom-control claim, as the proposed pipeline achieves both the highest Bloom scores and the most consistent evaluator agreement on inferred cognitive levels.

Agreement is generally lower for SBQs than SAQs, reflecting the added subjectivity of scenario relevance, grounding, and context-dependent grading. Criterion-level results further clarify where

Pipeline	Generator	SAQ Win Rate		SBQ Win Rate	
		Raw	Production	Raw	Production
ZS	GPT-4o	0.42 [0.30, 0.54]	0.47 [0.35, 0.58]	0.40 [0.20, 0.62]	0.17 [0.07, 0.28]
	Gemini	0.48 [0.29, 0.67]	0.27 [0.10, 0.46]	0.46 [0.34, 0.59]	0.34 [0.24, 0.44]
	GPT-4.1	0.39 [0.25, 0.51]	0.30 [0.20, 0.41]	0.41 [0.31, 0.50]	0.32 [0.23, 0.43]
ZS+B	GPT-4o	0.41 [0.30, 0.52]	0.28 [0.19, 0.38]	0.50 [0.38, 0.62]	0.42 [0.23, 0.62]
	Gemini	0.46 [0.35, 0.56]	0.38 [0.23, 0.52]	**0.60** [0.45, 0.75]	0.40 [0.25, 0.55]
	GPT-4.1	0.42 [0.30, 0.55]	0.47 [0.35, 0.58]	0.60 [0.47, 0.75]	0.41 [0.31, 0.53]
MZS	GPT-4o	0.23 [0.14, 0.31]	0.31 [0.21, 0.42]	0.42 [0.20, 0.62]	0.38 [0.20, 0.57]
	Gemini	0.29 [0.12, 0.50]	0.27 [0.15, 0.42]	0.47 [0.30, 0.65]	0.40 [0.28, 0.54]
	GPT-4.1	0.28 [0.19, 0.39]	0.27 [0.18, 0.38]	0.30 [0.19, 0.42]	0.32 [0.21, 0.45]
MZS+B	GPT-4o	0.44 [0.32, 0.57]	0.46 [0.35, 0.57]	0.55 [0.28, 0.80]	0.68 [0.47, 0.88]
	Gemini	0.46 [0.27, 0.65]	0.58 [0.48, 0.67]	0.35 [0.20, 0.50]	0.47 [0.31, 0.65]
	GPT-4.1	0.47 [0.35, 0.57]	0.47 [0.35, 0.57]	0.44 [0.31, 0.56]	0.52 [0.41, 0.63]
Ours	GPT-4o	**0.95** [0.89, 0.99]	**0.97** [0.92, 1.00]	**0.62** [0.40, 0.85]	**0.78** [0.65, 0.88]
	Gemini	**0.79** [0.56, 0.98]	**0.94** [0.88, 1.00]	**0.60** [0.44, 0.76]	**0.86** [0.78, 0.94]
	GPT-4.1	**0.92** [0.83, 0.99]	**0.94** [0.89, 0.99]	**0.75** [0.62, 0.86]	**0.92** [0.85, 0.97]

Table 5: Per-generator matched-task win rates with 95% bootstrap confidence intervals for SAQs and SBQs under raw and production selection modes. Win rates are computed against the pool of all other pipelines using discrimination-weighted scores aggregated across the three evaluator models. Raw mode uses the first generated candidate; production mode selects the highest-scoring candidate. GPT-4o denotes GPT-4o-mini; Gemini denotes Gemini-2.5-Flash. **Bold** = best per generator and evaluation mode; underline = second best.

agreement is most stable (Table 4). For SAQs, judges agree most strongly on clarity (0.954), course grounding (0.931), and answer fidelity (0.868), while MSQ has the highest pairwise correlation ($r = 0.740$). For SBQs, answer fidelity has the highest agreement (0.925), and MSQ again has the highest pairwise correlation ($r = 0.723$). By contrast, SRN has the lowest SBQ agreement (0.321), confirming that scenario necessity is the least stable judgement dimension.

Pairwise Outcomes Matched-task pairwise comparisons support the same overall trend (Table 5). For SAQs, the proposed pipeline achieves the highest win rates under every generator and selection mode. In raw mode, it obtains win rates of 0.95, 0.79, and 0.92 for GPT-4o-mini, Gemini-2.5 Flash, and GPT-4.1 respectively. In production mode, these increase to 0.97, 0.94, and 0.94. This indicates that the proposed pipeline improves both intrinsic generation quality and the quality of the criterion-based selected output.

For SBQs, the proposed pipeline achieves the strongest production-mode results, with win rates of 0.78, 0.86, and 0.92 across the three generators. Raw SBQ performance is less decisive, with win rates of 0.62, 0.60, and 0.75, reflecting the variability introduced by SBQs. The stronger production-mode results suggest that future deployment could benefit from criterion-based candidate selection, particularly for SBQs, where scenario-conditioned generation introduces greater variability.

Finally, the proposed pipeline exhibits a yield advantage in the retained item pool, with 1072 SAQs and 750 SBQs compared with the largest baseline pools of 261 and 246, respectively. Together with its stronger Bloom-control results, this suggests that the framework produces more valid assessment tuples while producing stronger alignment with the intended cognitive levels.

5 Conclusion

We presented a multi-agent, multi-stage framework for generating structured assessment items with Bloom-level cognitive control and automated verification. Across five pipelines, the results show that prompt-level Bloom conditioning improves alignment but remains less reliable than Bloom-specialized few-shot generation with staged verification. The proposed pipeline achieves the strongest overall performance, particularly in Bloom alignment, higher-order cognitive targeting, mark scheme quality, and valid-item yield. The results also show that SBQs remain more challenging than SAQs: scenario grounding improves, but scenario necessity, objective gradability, and answer fidelity are harder to satisfy consistently. Overall, reliable assessment generation appears to require pipeline-level specialization and validation rather than generic zero-shot or prompt-level Bloom conditioning alone.

Limitations

Our evaluation is conducted on a single domain: university-level Artificial Intelligence course material. Although the framework is intended to be domain-agnostic, we have not verified whether the same results generalize to disciplines with substantially different assessment conventions, such as medicine, law, humanities, or mathematically intensive subjects. Mathematical, numerical, proof-based, and programming-oriented questions may also require stricter verification than the rubric-based checks used here, since correct answers may involve deterministic values, multiple valid solution paths, or executable code. Therefore, future work should evaluate these item types separately, potentially combining rubric-guided judging with symbolic checking, unit tests, code execution, or answer-equivalence verification.

Although we complement automated evaluation with structured blind human assessment, the manual study covers only 18 matched-task comparisons across five pipelines. This provides useful qualitative validation, but limits the statistical conclusions that can be drawn. A larger-scale human evaluation with multiple annotators and inter-annotator agreement measures would more robustly validate the pedagogical quality of the generated items. Our automated evaluation uses three LLM-as-a-judge models rather than a single judge, but remains an imperfect proxy for pedagogical quality. The agreement analysis shows that Bloom-related measures and mark scheme quality are relatively more stable, whereas scenario relevance and necessity (SRN) are less stable. Thus, conclusions about Bloom control and mark scheme quality are better supported than conclusions about whether a scenario is genuinely necessary for answering an SBQ. Future work should refine the SRN rubric and validate this criterion more extensively with human assessors.

We evaluate proprietary API-based models only, using GPT-4.1, GPT-4o-mini, and Gemini-2.5 Flash as generators, and GPT-4o-mini, GPT-4.1-mini, and Gemini-2.5 Flash as evaluators. These models were chosen as a practical cost–efficiency trade-off, since the evaluation required scoring many generated items across multiple criteria. However, the framework is not tied to proprietary models. Future work should evaluate open-source generators and judges, including smaller models, to study model-size scaling, cost–performance trade-offs, and deployment in resource-constrained educational settings. Stronger evaluator models may also improve verification for reasoning-heavy or domain-specialist assessment items.

The generation and evaluation models are not fully independent, since GPT-family models appear in both roles, and Gemini is used as both a generator and evaluator. We reduce this risk by reporting results across multiple generator–evaluator configurations and including models from more than one provider, but a broader judge ensemble would further reduce the risk of model-family bias. We also do not explore heterogeneous pipeline configurations, such as using different models for question generation, scenario generation, mark scheme generation, answer generation, and verification.

The proposed pipeline is evaluated as an integrated framework, rather than as a controlled test of any individual component. It differs from the baselines through a combination of Bloom-specialized few-shot prompting, level-specific mark scheme templates, staged generation, scenario critique and refinement for SBQs, and generation-time verification. These choices are intended to work together, but the present experiments do not isolate their individual contributions. Future work should conduct component-level ablations, such as removing in-context examples, disabling verification, replacing Bloom-specialized prompts with generic Bloom prompts, or removing Bloom-aware mark scheme templates. Full specifications of each pipeline component are provided in Appendix A.

Finally, the yield advantage of the proposed pipeline comes from generating candidates for multiple Bloom levels and filtering them through verification, which increases computational cost. The framework does not use open-ended agent interaction. Each item passes through a fixed sequence of generation: critique, refinement, and verification stages, with failed candidates being discarded rather than recursively regenerated until they pass. The SAQ and SBQ acceptance predicates, including Boolean decision variables, JSON decision fields, and check logic, are specified in Appendix A.6. This design prevents unbounded verifier–generator loops and keeps execution predictable, but lowers the potential yield from the provided source material. We also do not analyse latency, cost, verifier failure rates, or alternative retry strategies in detail. Future work should study these trade-offs more systematically. The validated assessment tuples produced by the framework could also provide supervision for fine-tuning models that generate Bloom-aligned items directly from source context, potentially reducing the need for expensive multi-stage generation at inference.

References

Kuang Wen Chan, Farhan Ali, Joonhyeong Park, Kah Shen Brandon Sham, Erdalyn Yeh Thong Tan, Francis Woon Chien Chong, Kun Qian, and Guan Kheng Sze. 2025. Automatic item generation in various STEM subjects using large language model (LLM) prompting. *Computers and Education: Artificial Intelligence*, 8:100344.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Preprint*, arXiv:2507.06261.

Bidyut Das, Mukta Majumder, Santanu Phadikar, and Arif Ahmed Sekh. 2021. Automatic question generation and answer assessment: a survey. *Research and Practice in Technology Enhanced Learning*, 16(1):5.

Weiping Fu, Bifan Wei, Jianxiang Hu, Zhongmin Cai, and Jun Liu. 2024. Qgeval: Benchmarking multi-dimensional evaluation for question generation. *Preprint*, arXiv:2406.05707.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. 2024. Llm-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 13806–13834. Association for Computational Linguistics.

Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Solafa Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30:1–70.

Jiazheng Li, Artem Bobrov, David West, Cesare Aloisi, and Yulan He. 2024a. An automated explainable educational assessment system built on llms. *Preprint*, arXiv:2412.13381.

Jiazheng Li, Lin Gui, Yuxiang Zhou, David West, Cesare Aloisi, and Yulan He. 2023. Distilling ChatGPT for explainable automated student answer assessment. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6007–6026, Singapore. Association for Computational Linguistics.

Jiazheng Li, Hainiu Xu, Zhaoyue Sun, Yuxiang Zhou, David West, Cesare Aloisi, and Yulan He. 2024b. Calibrating LLMs with preference optimization on thought trees for generating rationale in science question scoring. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5452–5479, Miami, Florida, USA. Association for Computational Linguistics.

Jiazheng Li, Yuxiang Zhou, Junru Lu, Gladys Tyen, Lin Gui, Cesare Aloisi, and Yulan He. 2025. Two heads are better than one: Dual-model verbal reflection at inference-time. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3119–3140, Suzhou, China. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *Preprint*, arXiv:2303.16634.

Anderson LW, Krathwohl DR, Airasian PW, Cruikshank KA, Richard Mayer, Pintrich PR, J. Raths, and Wittrock MC. 2001. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

Subhankar Maity, Aniket Deroy, and Sudeshna Sarkar. 2024. How effective is gpt-4 turbo in generating school-level questions from textbooks based on bloom's revised taxonomy? *Preprint*, arXiv:2406.15211.

Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. 2023. Rquge: Reference-free metric for evaluating question generation by answering the question. *Preprint*, arXiv:2211.01482.

Bang Nguyen, Mengxia Yu, Yun Huang, and Meng Jiang. 2024. Reference-based metrics disprove themselves in question generation. *Preprint*, arXiv:2403.12242.

Shinhyeok Oh, Hyojun Go, Hyeongdon Moon, Yunsung Lee, Myeongho Jeong, Hyun Seung Lee, and Seungtaek Choi. 2023. Evaluation of question generation needs more references. *Preprint*, arXiv:2305.16626.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Nicy Scaria, Suma Dharani Chenna, and Deepak Subramani. 2024. Automated educational question generation at different bloom's skill levels using large language models: Strategies and evaluation. In *Proceedings of the 25th International Conference on Artificial Intelligence in Education (AIED)*. ArXiv preprint arXiv:2408.04394.

Y. Song, J. Du, and Q. Zheng. 2025. Automatic item generation for educational assessments: a systematic literature review. *Interactive Learning Environments*, 33(9):5386–5405.

Bin Tan, Nour Armoush, Elisabetta Mazzullo, Okan Bulut, and Mark Gierl. 2025. A review of automatic item generation techniques leveraging large language models (LLMs). *International Journal of Assessment Tools in Education*, 12(2):317–340.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and 1332 others. 2025. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Masaki Uto, Yuto Tomikawa, and Ayaka Suzuki. 2023. Difficulty-controllable neural question generation for reading comprehension using item response theory. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 119–129, Toronto, Canada. Association for Computational Linguistics.

Antoun Yaacoub, Jérôme Da-Rugna, and Zainab Assaghir. 2025. Assessing ai-generated questions' alignment with cognitive frameworks in educational assessment. *International Journal of Computer Theory and Engineering*, 17(3):114–125.

Archana Yadav, Harshvivek Kashid, Medchalimi Sruthi, B JayaPrakash, Chintalapalli Raja Kullayappa, Mandala Jagadeesh Reddy, and Pushpak Bhattacharyya. 2025. From recall to creation: Generating follow-up questions using bloom's taxonomy and grice's maxims. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1322–1338, Vienna, Austria. Association for Computational Linguistics.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. 2021. A review on question generation from natural language text. *ACM Trans. Inf. Syst.*

Runcong Zhao, Artem Bobrov, Jiazheng Li, Cesare Aloisi, and Yulan He. 2025. LearnLens: LLM-enabled personalised, curriculum-grounded feedback with educators in the loop. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 625–633, Suzhou, China. Association for Computational Linguistics.

Code repository. Code and supporting materials for the generation and evaluation pipeline are available at: `https://github.com/LyleW473/BloomAssessmentItemGeneration`.

A Generation Pipeline Details

A.1 Pipelines & Prompt Templates

This section describes the generation-time pipeline used to produce assessment items. All pipelines produce structured JSON outputs and return generated items as either `valid_questions` or `failed_questions`. Across the proposed pipeline, difficulty is not independently assigned by the model; instead, it is deterministically derived from the target Bloom level: `knowledge` and `understanding` map to `easy`, `application` and `analyze` map to `medium`, and `synthesis` and `evaluation` map to `hard`.

A.2 Baseline Generation Pipelines

The zero-shot baseline (ZS) generates the full assessment tuple in a single LLM call. For SAQs, the model outputs a question, difficulty, mark scheme, total marks, and expected answer. For SBQs, the same call additionally generates a scenario. The zero-shot Bloom baseline (ZS+B) uses the same single-stage structure, but augments the system prompt with a Bloom taxonomy definition block and asks the model to assign a `bloom_level` to each item.

The multi-stage zero-shot baseline (MZS) decomposes generation into three sequential calls: question generation, mark scheme generation, and expected answer generation. For SBQs, the first stage generates both the scenario and the question. The multi-stage Bloom baseline (MZS+B) adds Bloom taxonomy definitions to the first-stage question generation prompt and records the model-assigned `bloom_level`; however, this Bloom label does not directly control the later mark scheme or answer generation stages.

A.3 Proposed Generation Pipeline

The proposed pipeline differs from the baselines in three main ways. First, generation is driven by the target Bloom level rather than relying on the model to self-assign a level. The pipeline iterates over all six Bloom levels and invokes a level-specific question-generation prompt for each level. Second, these Bloom-level prompts are few-shot templates, each containing level-specific instructions, allowed and prohibited question forms, output-format constraints, and worked examples illustrating the intended cognitive demand. In contrast, the baseline prompts specify task instructions and output schemas, but do not include content-filled in-context demonstrations. Third, generation

is interleaved with verification, where candidates that fail Bloom alignment, grounding, mark scheme coverage, or answer coverage checks are discarded rather than passed to later stages.

For SAQs, the proposed pipeline follows the structure:

- For each Bloom level $b \in \mathcal{B}$:
  - Generate question stems from the source content using the prompt for b.
  - Verify Bloom alignment and course relevance.
  - Generate a Bloom-aware mark scheme.
  - Verify mark scheme coverage.
  - Generate the expected answer.
  - Verify answer coverage and keep valid items.

For SBQs, the proposed pipeline follows the structure:

- Generate an initial scenario from the source content.
- Critique the scenario for compliance.
- Refine the scenario if required.
- For each Bloom level $b \in \mathcal{B}$:
  - Generate SBQ stems from the refined scenario using the prompt for b.
  - Verify Bloom alignment and scenario relevance.
  - Generate a Bloom-aware mark scheme.
  - Verify mark scheme coverage with respect to the scenario.
  - Generate the expected answer.
  - Verify answer coverage and keep valid items.

The SBQ pipeline therefore extends the SAQ tuple-generation process with a scenario generation–critique–refinement stage. After refinement, the scenario is used as the factual input for Bloom-level question generation, mark scheme generation, answer generation, and verification. This means that SBQs use the same Bloom-controlled generation logic as SAQs, but all downstream stages are conditioned on the generated scenario rather than directly on the source text.

A.4 Bloom-Level Question Generation Prompts

In the proposed pipeline, question generation is controlled by separate Bloom-level system prompts rather than a single generic Bloom instruction. Each Bloom level has its own prompt, but all prompts follow the same template structure. The prompt first assigns the model the role of an expert educational content generator specializing in a specific Bloom level, then defines the cognitive operation that questions at that level should require. It then specifies the input format, output JSON schema, allowed command verbs, prohibited lower- or higher-level stems, example question forms, and strict output formatting requirements.

Bloom Level	Cognitive Goal	Representative Verbs
Knowledge	Recall facts and terms	define, identify, list, state
Understanding	Explain or interpret meaning	explain, summarize, interpret, illustrate
Application	Use knowledge in a new case	apply, compute, solve, demonstrate
Analyze	Break down relationships	analyze, compare, differentiate, infer
Evaluation	Make justified judgements	assess, defend, evaluate, justify
Synthesis	Construct a new whole	design, formulate, generate, plan

Table 6: Summary of the Bloom-level prompt specialization used in the proposed pipeline. Each level uses a dedicated prompt with level-specific goals, allowed verbs, prohibited stems, and examples.

The generic structure of each Bloom-level prompt is:

- **Role:** Assigns the model the role of an expert generator for a specific Bloom level.
- **Goal:** Defines the target cognitive operation required by that Bloom level.
- **Input:** Specifies the source content and target Bloom level.
- **Output:** Requires a JSON array of question objects.
- **Guidelines:** Lists allowed verbs and the expected reasoning depth.
- **Restrictions:** Prohibits off-level stems and lower- or higher-level phrasing.
- **Examples:** Provides level-specific examples of valid question forms.
- **Formatting:** Requires valid JSON only, with no additional commentary.

Thus, the same Bloom-specialized prompt structure is used for both formats, but SAQs generate questions directly from course content, while SBQs generate questions from the refined scenario.

The SBQ application-level prompt includes an additional eligibility constraint. An application-level SBQ is generated only if the scenario contains sufficient numerical values, counts, or structured outcomes to support the application of methods. If this information is absent, the model is instructed to return an empty array. This prevents computationally framed questions from being generated when the scenario does not contain the information necessary to solve them.

Bloom Level	Points	Marks/Point	Total Marks
Knowledge	1–3	1	1–3
Understanding	2–4	1–2	3–5
Application	3–5	1–2	4–8
Analyze	4–6	1–2	6–10
Evaluation	4–6	1–2	6–10
Synthesis	5–8	1–3	8–12

Table 7: Bloom-level mark scheme templates used by the proposed pipeline. The model must output a single integer mark value for each point; range strings such as `"1-2"` are not allowed.

A.5 Bloom-Aware Mark Scheme Generation

The proposed pipeline also uses a Bloom-aware few-shot mark scheme prompt. Unlike the baseline mark scheme prompts, which provide only general marking guidance and schema constraints, this prompt conditions the structure of the mark scheme on the target Bloom level and includes worked examples showing how marking points should differ across cognitive levels. The input specifies the question, difficulty, Bloom level, and source content, and the model is instructed to return a structured JSON mark scheme with one integer mark value per point.

The prompt constrains both the *structure* and *content* of the mark scheme. Structurally, each Bloom level is assigned an expected range of marking points, marks per point, and total marks. Content-wise, lower Bloom levels require concise factual marking points, while higher Bloom levels require points covering reasoning, comparison, judgement, design components, or justification. Thus, the mark scheme is not generated as a generic list of possible answer points, but is explicitly shaped by the intended cognitive demand of the question through both level-specific constraints and few-shot exemplars.

The prompt further specifies level-specific content expectations:

- **Knowledge**: Strictly factual points with no elaboration.
- **Understanding**: Light explanatory points without deep reasoning.
- **Application**: Concrete procedural or method-application steps.
- **Analyze**: Structural, relational, or comparative points.
- **Evaluation**: Criteria-based judgement points.
- **Synthesis**: Design components, justification, and rationale.

Difficulty affects the expected depth of each marking point, but not the Bloom-level point-count or mark-allocation template. Easy items require minimal depth, medium items require moderate interconnected detail, and hard items require more precise technical explanation. After generation, the implementation recomputes `total_marks` as the sum of the generated point values, rather than relying on the model-provided total. This enforces consistency between the mark allocation and the reported total score, and reduces the number of valid candidates being rejected due to simple arithmetic errors in the generated total.

A.6 Generation-Time Verification Predicate

The proposed pipeline uses a verification predicate during generation to decide whether a candidate assessment tuple is retained. This generation-time filtering is distinct from the final evaluation protocol in Section B. Verification determines whether a candidate is passed into the generated dataset, whereas evaluation scores the items within the generated dataset.

All verification checks are implemented as binary rubric-guided LLM calls that return structured JSON.

Design principle. There are no numeric generation-time thresholds. Each verifier returns a strict Boolean decision via a named JSON field (see Table 8), and a single `False` at any stage rejects the candidate tuple, writing it to the failed-output pool. All checks are short-circuiting: once a candidate fails, no subsequent checks are run for that item.

For SAQs, a candidate tuple is accepted only if it passes all four checks:

$$\mathrm{ACCEPT}(q, m, a) \iff v_{\mathrm{Bloom}} \wedge v_{\mathrm{rel}} \wedge v_{\mathrm{ms}} \wedge v_{\mathrm{ans}},$$

where q is the question, m the mark scheme, and a the expected answer. The four decision variables correspond to Bloom-level alignment, course relevance, mark scheme coverage, and expected answer coverage.

For SBQs, the same tuple-level predicate is applied after an additional scenario-compliance stage. A scenario s is first generated, critiqued for compliance, and conditionally refined to produce a final scenario s'. The candidate SBQ tuple is then accepted only if the question, mark scheme, and expected answer pass the same four checks with respect to s':

$$\mathrm{ACCEPT}_{\mathrm{SBQ}}(s', q, m, a) \iff v_{\mathrm{Bloom}} \wedge v^{s}_{\mathrm{rel}} \wedge v^{s}_{\mathrm{ms}} \wedge v_{\mathrm{ans}}.$$

Check	Input	Decision field	Failure action
Bloom alignment	Question, target Bloom level	`belongs_to_level`	Reject
Course/scenario relevance	Question, source content or scenario	`is_relevant`	Reject
Mark scheme coverage	Question, mark scheme, Bloom level, content	`is_correct`	Reject
Expected answer coverage	Question, mark scheme, expected answer	`is_covered`	Reject

Table 8: Summary of generation-time verification checks. All checks return a Boolean decision and a short reason.

Bloom level	Difficulty
Knowledge	easy
Understanding	easy
Application	medium
Analyze	medium
Evaluation	hard
Synthesis	hard

Table 9: Deterministic mapping from target Bloom level to item difficulty.

Here, v^s_{rel} verifies that the question is grounded in the refined scenario, and v^s_{ms} verifies that the mark scheme is also grounded in that scenario. Thus, SBQs use the same acceptance logic as SAQs, but with an additional scenario generation–critique–refinement stage and scenario-grounded relevance checks.

Difficulty is derived deterministically from the target Bloom level rather than assigned independently. We use the mapping in Table 9 for mark scheme generation and subsequent verification.

SAQ verification. For each source document, the SAQ pipeline iterates over all Bloom levels. It first generates question stems using Bloom-specialized system instructions. Each stem is then verified for Bloom alignment and course relevance. Only stems passing both phase-one checks proceed to mark scheme generation. The generated mark scheme is then checked for coverage, factual correctness, Bloom-level structural compliance, and alignment with the question. If this check passes, an expected answer is generated and verified against the mark scheme. Only candidates passing all four checks are added to the valid-output pool.

The Bloom alignment check asks the verifier to infer the true cognitive level of the question from its phrasing and cognitive demand, then compare it with the declared target level. The course relevance check verifies that the question can be answered from substantive course content rather than from administrative or other external information. The mark scheme coverage check verifies that the marking points are factually accurate, complete, aligned with the question, and consistent with the required Bloom-level structure. The expected answer coverage check verifies that the prose answer expresses every mark scheme point at the required cognitive depth without contradiction.

Bloom level	Points	Marks/point	Total marks
Knowledge	1–3	1	1–3
Understanding	2–4	1–2	3–5
Application	3–5	1–2	4–8
Analyze	4–6	1–2	6–10
Evaluation	4–6	1–2	6–10
Synthesis	5–8	1–3	8–12

Table 10: Bloom-level structural templates used for mark scheme generation and verification.

SBQ verification. The SBQ pipeline adds a scenario-compliance phase before the four question-level checks. First, a neutral factual scenario is generated from the source content. A critique agent then reviews the scenario for seven violation types: metric-as-conclusion, teaching or definition, interpretation or judgement, mental-state or goal language, rationale or intent, recommendation or next step, and question leakage. If the scenario is compliant, it is used directly. If not, a refinement step applies the minimal changes required to remove the detected violations.

After scenario refinement, SBQ question generation and verification proceed as in the SAQ pipeline. The key difference is that relevance and mark scheme grounding are checked against the refined scenario rather than the full source document. This ensures that the question and mark scheme are grounded in the scenario actually presented to the student.

The framework therefore uses a fixed-stage verification process rather than open-ended agent interaction. Failed question or tuple candidates are discarded rather than recursively regenerated until they pass, which keeps execution predictable and prevents unbounded verifier–generator loops. However, this design also means that the system may sacrifice potential yield from source material when otherwise recoverable candidates fail a verification stage.

B Evaluation Details

B.1 Bloom-Specific Metrics

Let $\hat{b}_i$ denote the inferred Bloom level of generated item i, and b_i its corresponding target level. For a given pipeline p and generation model g, Bloom alignment accuracy is defined as

$$\text{BA}^{(p,g)} = \frac{1}{|\mathcal{D}^{(p,g)}|} \sum_{i \in \mathcal{D}^{(p,g)}} I(\hat{b}_i = b_i),$$

which measures the proportion of generated questions whose inferred cognitive level matches the intended one.

To isolate the challenge of generating higher-order questions, we restrict evaluation to $\mathcal{B}_{\text{high}} = \{\texttt{analysis}, \texttt{evaluation}, \texttt{synthesis}\}$ and define the higher-order success rate as

$$\text{HOSR}^{(p,g)} = \frac{1}{|\mathcal{D}_{\text{high}}^{(p,g)}|} \sum_{i \in \mathcal{D}_{\text{high}}^{(p,g)}} I(\hat{b}_i = b_i),$$

where $\mathcal{D}_{\text{high}}^{(p,g)}$ denotes the subset of questions whose target Bloom level lies in $\mathcal{B}_{\text{high}}$.

B.2 Criterion-Based Scoring

Given a pipeline $p \in \mathcal{P}$, a generation model $g \in \mathcal{G}$, and a set of evaluation criteria $\mathcal{C}$, each generated item i is assigned a discrete score $s_{i,c}^{(p,g)} \in \{0, 1, 2\}$ for criterion $c \in \mathcal{C}$ using a rubric-guided LLM judge. Scores are normalized to the unit interval as $\tilde{s}_{i,c}^{(p,g)} = s_{i,c}^{(p,g)}/2$.

Aggregate performance is characterized by the mean normalized score per criterion:

$$\mu_c^{(p,g)} = \frac{1}{N_{p,g}} \sum_{i=1}^{N_{p,g}} \tilde{s}_{i,c}^{(p,g)}.$$

B.3 Discrimination-Weighted Aggregation

To obtain a single scalar score per item, we introduce a discrimination-weighted aggregation scheme that emphasizes criteria which better separate pipeline performance.

For each criterion $c \in \mathcal{C}$, we first compute the mean score per pipeline by averaging across generation models:

$$\mu_c^{(p)} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \mu_c^{(p,g)}.$$

The discrimination power of criterion c is then defined as

$$d_c = \max_{p \in \mathcal{P}} \mu_c^{(p)} - \min_{p \in \mathcal{P}} \mu_c^{(p)}.$$

To stabilize the contribution of high-variance criteria, we apply a square-root transformation $\tilde{d}_c = \sqrt{d_c}$ and normalize:

$$w_c = \begin{cases} \frac{\tilde{d}_c}{\sum_{c' \in \mathcal{C}} \tilde{d}_{c'}} & \text{if } \sum_{c'} \tilde{d}_{c'} > 0, \\ \frac{1}{|\mathcal{C}|} & \text{otherwise.} \end{cases}$$

The final score for a generated item y with normalized criterion scores $\{\tilde{s}_c\}_{c \in \mathcal{C}}$ is

$$S(y) = \sum_{c \in \mathcal{C}} w_c \, \tilde{s}_c.$$

B.4 Matched-Task Pairwise Comparison

A task slot is defined as a pair (f, d), where f denotes the source file and d the target Bloom level. Comparisons are restricted to task slots where both pipelines produced outputs. Each comparison unit is a matched triple (f, d, g).

For each matched triple, let $\mathcal{Y}_p^{(f,d,g)}$ denote the set of candidate outputs from pipeline p. In *raw* mode, the representative is the first generated candidate. In *production* mode, the representative is the candidate with the highest weighted score:

$$y_p^{(f,d,g)} = \arg \max_{y \in \mathcal{Y}_p^{(f,d,g)}} S(y).$$

With $S_1 = S(y_{p_1}^{(f,d,g)})$ and $S_2 = S(y_{p_2}^{(f,d,g)})$, the pairwise outcome is

$$\text{cmp}(p_1, p_2; f, d, g) = \begin{cases} \texttt{win}_{p_1} & \text{if } S_1 > S_2, \\ \texttt{win}_{p_2} & \text{if } S_2 > S_1, \\ \texttt{tie} & \text{otherwise.} \end{cases}$$

When weighted scores are equal, ties are resolved via a fixed lexicographic ordering over priority criteria. Outcomes are aggregated over all matched triples for each pipeline pair, and we apply bootstrap resampling over matched comparisons to report 95% confidence intervals for pairwise win rates.

C Additional Results

This section provides extended analysis of the experimental results through per-model breakdowns, outcome distributions, pairwise dominance matrices, Bloom-stratified comparisons, and score margin analysis.

High-Fidelity Compression of Course Content

A key challenge in scalable question generation is the prohibitive length and noise present in raw instructional materials. Directly prompting on full lecture notes, slides, and auxiliary documents introduces redundancy, administrative artifacts, and excessive input length that degrade both generation quality and throughput. To address this, we introduce a high-fidelity compression stage that reduces input size while preserving all pedagogically relevant content required for assessment generation.

Rather than performing conventional summarization, our approach enforces a *loss-constrained reduction* of the source material. Compression is implemented using a Gemini-3 Pro (Team et al., 2025) agent guided by a structured system prompt that explicitly distinguishes between examinable

and non-examinable content. The process retains all definitions, conceptual distinctions, procedural steps, examples, and technical terminology, while removing only administrative text, repeated explanations, and low-information filler. Importantly, the compression is conservative: original phrasing is preserved wherever possible, paraphrasing is minimized, and deduplication is applied only when semantic equivalence is clear. This ensures that the resulting representation remains faithful to the source material and suitable for downstream generation.

The compressed output is organized into coherent topical chunks, grouping related concepts, definitions, and examples. This structured representation improves prompt locality and enables more targeted question generation across Bloom levels. Additionally, lightweight standardization of mathematical expressions is applied to improve clarity without altering meaning or introducing new notation.

The compression pipeline is fully automated and includes validation to ensure structural consistency and completeness. For each document, we record the original and compressed word counts, percentage reduction, and an assigned content category. Only documents classified as *course_concept* are retained for generation, while administrative and adjacent materials are filtered out, ensuring that the generation stage operates exclusively on pedagogically relevant content.

As shown in Table 11, the proposed compression reduces total text length by 78.5% (measured in words) across the dataset, with comparable reductions for both retained and filtered subsets. Despite this substantial reduction, the preserved content remains sufficient to support high-quality question generation, demonstrating that aggressive input reduction can be achieved without compromising instructional fidelity.

Per-model win rates. Figure 3 breaks down matched-task win rates by generation model and selection mode. The proposed pipeline achieves the highest win rates across all model and setting combinations, with GPT-4o-mini showing the largest absolute win rates for SAQs in both raw and production modes. This suggests that the structured pipeline is particularly effective at compensating for the limitations of smaller models. For SBQs, the advantage is more uniform across models, with Gemini-2.5 Flash showing a slightly narrower margin, consistent with the observation that SBQ generation is inherently more variable.

Outcome distributions. Figure 4 presents the full win/tie/loss distributions averaged across models. The proposed pipeline not only achieves the highest win proportion in all settings, but also maintains the smallest loss proportion. In production mode, losses become negligible for SAQs (0.0%) and remain very low for SBQs (8.5%), confirming that candidate selection effectively eliminates the weakest outputs. Among baselines, Bloom-aware variants (ZS+B, MZS+B) show notably higher tie rates than their non-Bloom counterparts, suggesting that Bloom conditioning narrows the quality gap but does not consistently produce superior outputs.

Effect of production selection. Figure 5 isolates the effect of switching from raw to production mode. Production selection affects pipelines unevenly: for SAQs, it generally reduces win rates for weaker baselines while leaving the proposed pipeline largely unchanged, indicating that the proposed method already generates high-quality first candidates. For SBQs, production selection yields meaningful gains for the proposed pipeline and for MZS+B, with the largest improvements observed under Gemini-2.5 Flash. This asymmetry reflects the higher output variability in SBQ generation, where having multiple candidates and a quality-aware selection mechanism provides greater benefit.

Pairwise dominance. Figure 6 shows head-to-head win-rate matrices for all pipeline pairs. The proposed pipeline achieves win rates above 0.50 against every baseline in nearly all conditions, confirming broad dominance rather than gains concentrated against a single weak comparator. The matrices also reveal that among baselines, the relationships are more mixed: ZS+B and MZS+B trade wins depending on the format and selection mode, while non-Bloom variants (ZS, MZS) consistently underperform. This supports the interpretation that Bloom conditioning provides a meaningful baseline improvement, but the full proposed framework is needed for consistent superiority.

Bloom-level stratification. Figure 7 stratifies win rates by target Bloom level. The proposed pipeline shows advantages across most cognitive categories, with especially pronounced gains at the comprehension and analysis levels. For higher-order levels (evaluation and synthesis), the advantage is maintained but with smaller margins and more limited data, consistent with the known difficulty of generating and evaluating higher-order assessment items. Notably, at the Knowledge level, differences between pipelines are smaller, reflecting that factual

recall questions are comparatively straightforward for all methods.

Score margins. Figure 8 quantifies not just whether the proposed pipeline wins, but by how much. The proposed pipeline maintains positive mean score margins against all baselines in every setting. Margins are largest against ZS and MZS (the non-Bloom baselines), reflecting the combined benefit of cognitive control and structural verification. Against ZS+B and MZS+B, margins are smaller but consistently positive, indicating that the gains from Bloom specialization and verification are real but more incremental when the baseline already includes Bloom prompting. Per-model markers show that margins are relatively stable across generators, with no single model driving the aggregate advantage.

D Manual Qualitative Evaluation

To complement the automated criterion-based evaluation, we conduct a structured blind manual evaluation of matched-task sample outputs. A human evaluator assessed generated assessment items across both question formats and all difficulty levels, without knowledge of which pipeline produced which item.

D.1 Evaluation Protocol

Evaluations were conducted in two settings. *Representative evaluations* examine one matched comparison per difficulty level (easy, medium, hard) for both SAQs and SBQs, selecting the task slot that best represents the typical quality of each pipeline at that difficulty. *Illustrative contrast evaluations* select task slots specifically chosen to expose quality differences between pipelines along two axes: the overall quality gap (`max_score_gap`, i.e., the slot where the weighted score difference between the best and worst pipeline is greatest) and the Bloom alignment gap (`bloom_bla_gap`, i.e., the slot where the difference in Bloom alignment scores is greatest). All matched triples are presented to the evaluator blindly, with pipeline identities anonymized via random relabeling.

Each item is ranked against all others in its comparison set using the following criteria:

- **Bloom level alignment**: Consistency between declared and actual cognitive demand.
- **Course grounding**: Faithfulness to source material and correct use of concepts.
- **Clarity and specificity**: Unambiguous question wording with a well-defined task.
- **Single-task integrity**: The question assesses one coherent objective.
- **Objective gradability**: The mark scheme contains concrete, checkable marking points.
- **Mark scheme quality**: Coverage and logical structure of marking points.
- **Answer fidelity**: Alignment between the expected answer and the mark scheme.
- **Scenario relevance and necessity** (SBQs only): The scenario meaningfully contributes to the assessment task rather than providing merely decorative context.
- **Scenario grounding** (SBQs only): The scenario is realistic and consistent with course content.

D.2 Aggregate Results

Table 14 summarizes selection outcomes across all 18 comparisons. The proposed pipeline is ranked first in 13 of 18 settings (72.2%). Its advantage is most pronounced on Bloom-sensitive evaluations: it wins all three illustrative contrast comparisons targeting Bloom alignment gaps for SAQs, and two of three for SBQs. The only settings in which it does not rank first are the easy representative and easy illustrative contrast evaluations for SBQs, where Bloom-aware baselines benefit from the simpler generation demands of lower-difficulty items.

D.3 Representative Evaluations

In representative evaluations, one item per pipeline is drawn from the task slot most representative of each pipeline's typical quality at a given difficulty. Rankings are assigned by the evaluator based on the full assessment triple.

SAQs. Table 15 reports the full ranking across the three SAQ difficulty levels. At easy difficulty, the proposed pipeline ranks first with a precisely scoped Knowledge-level question that achieves exact Bloom alignment and a concise, fully matched mark scheme. The key separating factor is whether the declared cognitive level honestly reflects the actual task: the two lowest-ranked baselines (ZS, MZS) label their outputs as Knowledge-level but effectively require Understanding-level reasoning, a pattern consistently penalized across all evaluation settings.

At medium difficulty, the proposed pipeline again ranks first with a genuine Application-level item that embeds a target concept in a concrete, well-bounded problem. The second-ranked baseline (MZS+B) produces a strong Comparative Analysis question, but the remaining baselines are weakened by either Bloom misalignment or overly broad

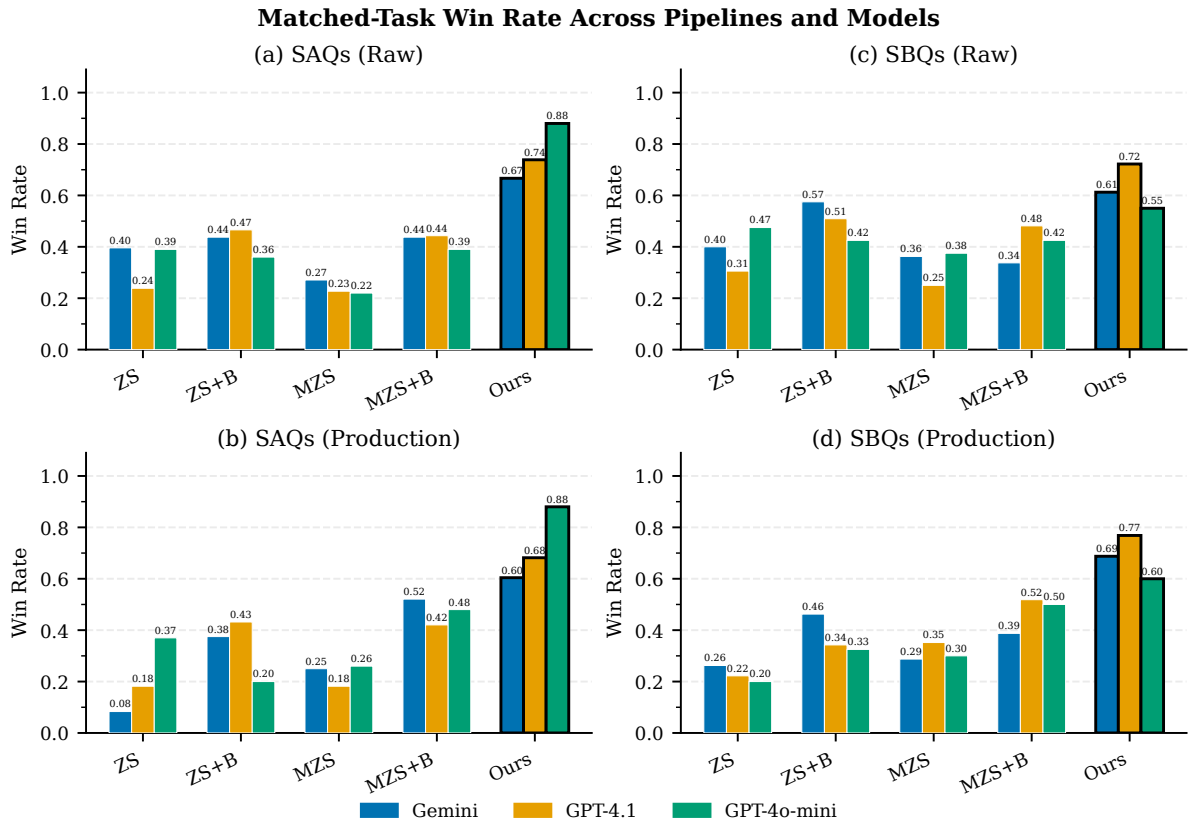

Figure 3: Matched-task win rates across pipelines, models, and selection settings for SAQs and SBQs. Tasks are matched by source content, target Bloom level, and generation model. Raw mode uses the first candidate; production mode selects the highest-scoring candidate.

question stems. At hard difficulty, MZS+B ranks first with a clean Evaluation question that requires justified decision-making under real constraints, while the proposed pipeline places second. The remaining baselines underdeliver at the Evaluation level by drifting toward Analysis or Understanding. Full annotated examples for easy, medium, and hard difficulty are shown in Figures 21, 22, and 23 respectively.

SBQs. Table 16 reports representative SBQ rankings. At easy difficulty, ZS+B ranks first with a well-grounded Understanding-level item that balances scenario relevance with mark scheme precision. The proposed pipeline ranks third: its item is structurally sound and Bloom-aligned, but the scenario is disproportionately large relative to the narrow scope of the question being asked, weakening the assessment's use of contextual framing. MZS places last across all three difficulty levels.

At medium difficulty, the proposed pipeline ranks first with an internally consistent Analyze-level question that uses the scenario directly and necessarily to support a structured comparative Analysis task. At hard difficulty, the proposed pipeline again ranks first with a Synthesis-level item requiring students to develop a principled guidelines framework from a complex AI ethics case study. The separating factors across all three levels are correct Bloom alignment and proportionate, necessary scenario use. Full annotated examples for easy, medium, and hard difficulty are shown in Figures 24, 25, and 26 respectively.

D.4 Illustrative Contrast Evaluations

Illustrative contrast evaluations use task slots specifically selected to expose qualitative differences between pipelines. Each difficulty level contributes one slot per axis: `max_score_gap` for overall quality spread and `bloom_bla_gap` for Bloom alignment spread.

SAQ illustrative contrasts. Tables 17 and 18 report rankings on the `max_score_gap` and `bloom_bla_gap` axes respectively. On the `max_score_gap` axis, the proposed pipeline ranks first in four of six settings. At easy difficulty it tops the ranking with a precisely defined Knowledge-level CSP item exhibiting exact Bloom alignment and a minimal, coherent mark scheme. The single

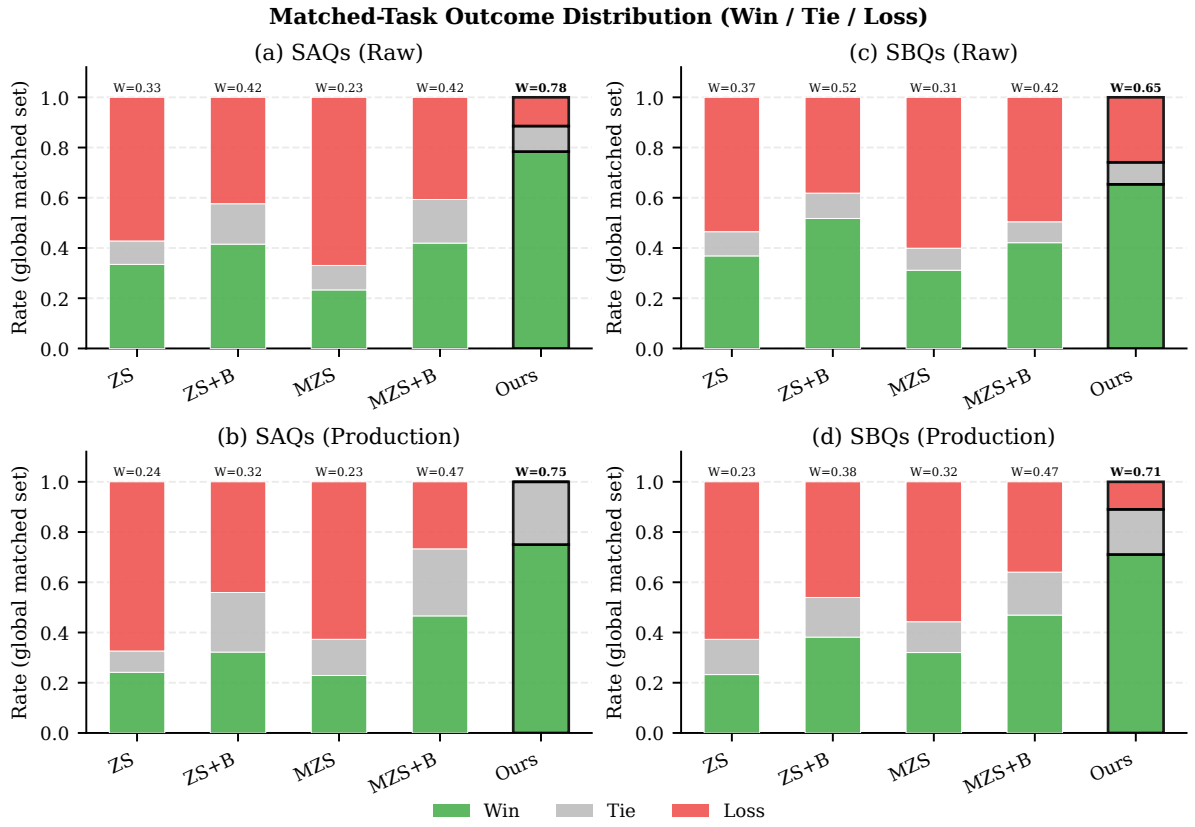

Figure 4: Stacked win/tie/loss distributions per pipeline, averaged across models, on the global matched-task set. Across SAQs and SBQs in both raw and production settings, the proposed pipeline achieves the most favourable outcome distribution, with the highest win rates and near-zero loss rates in production.

`max_score_gap` setting it does not win is medium difficulty, where ZS+B produces a clean comparative Analysis question judged to be more naturally framed and better pitched at that cognitive level. Full annotated examples are shown in Figures 12, 13, and 14.

On the `bloom_bla_gap` axis, the proposed pipeline ranks first in all three settings. In every case, the primary discriminating factor is whether the declared Bloom level accurately reflects the actual cognitive demand: lower-ranked items are penalized either for labeling Understanding-level questions as Knowledge or for inflating Evaluation labels onto questions that are more naturally Understanding or Application. The proposed pipeline's combination of Bloom-specialized generation and explicit verification is directly responsible for this consistent advantage. Full annotated examples are shown in Figures 9, 10, and 11.

SBQ illustrative contrasts. Tables 19 and 20 report SBQ contrast rankings. On the `max_score_gap` axis, MZS+B ranks first in both easy settings while the proposed pipeline wins medium and hard. This pattern is consistent with the representative results: on straightforward easy tasks where scenario grounding requirements are less demanding, Bloom-aware baselines can match or exceed the proposed pipeline. At medium difficulty the proposed pipeline tops the ranking with the most precisely controlled and objectively gradable application item, and at hard difficulty with the strongest evaluative judgment task. Full annotated examples are shown in Figures 18, 19, and 20.

On the `bloom_bla_gap` axis, MZS+B again ranks first at easy difficulty. At medium and hard, the proposed pipeline ranks first with items whose scenarios are essential to the reasoning being asked and whose declared cognitive level accurately matches the actual task. The recurring failure pattern in lower-ranked baselines is scenario-question mismatch: either a large elaborate scenario is provided for a question that barely uses it, or the question substantially overstates the cognitive complexity relative to the declared Bloom label. Full annotated examples are shown in Figures 15, 16, and 17.

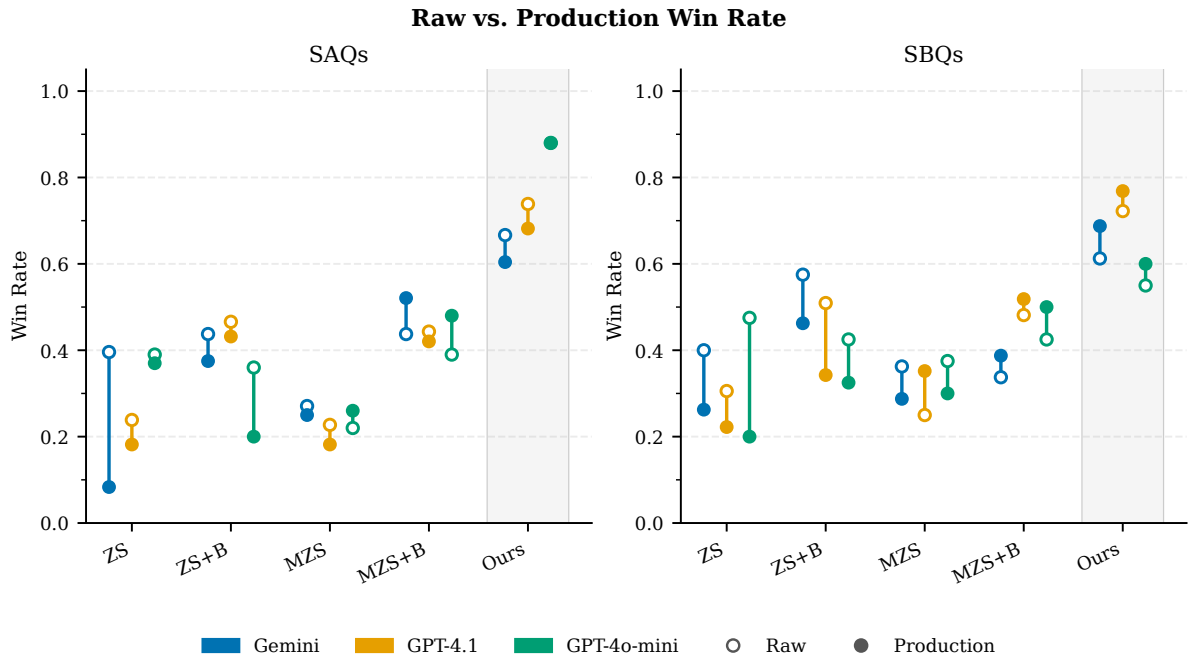

Figure 5: Per-model raw (hollow) and production (filled) win rates across pipelines. The vertical gap between markers shows the effect of production selection, which is modest for SAQs but clearer for SBQs, especially for the stronger pipelines.

D.5 Key Qualitative Observations

Several consistent patterns emerge across all evaluation settings.

Bloom alignment as primary differentiator. The single most consequential factor across all qualitative evaluations is whether the declared Bloom level accurately reflects the actual cognitive demand of the question. Items labeled as Knowledge but requiring Understanding-level reasoning, or labeled as Evaluation while only demanding Explanation, are consistently ranked below correctly aligned alternatives regardless of surface-level quality. This pattern is most salient in the `bloom_bla_gap` contrast evaluations, where the proposed pipeline wins all three SAQ settings and two of three SBQ settings. Lower-ranked baselines systematically exhibit one of two failure modes: cognitive understatement (e.g., labeling Analysis as Knowledge) or cognitive inflation (e.g., labeling Understanding as Evaluation), the latter being particularly common at hard difficulty.

Mark scheme coherence as separating criterion. Even when question stems are well formed, differences in mark scheme quality reliably separate pipeline ranks. The strongest items have mark schemes whose scope exactly matches the question stem, with concrete, independently verifiable marking points. Weaker items either over-extend — including points not implied by the stem — or under-specify — bundling multiple related concepts into a single marking point — making consistent grading more difficult. This accounts for several mid-range rankings at medium and hard difficulty where the stem is strong but the mark scheme introduces uncertainty.

Scenario use in SBQs. For scenario-based questions, a recurring weakness in lower-ranked outputs is mismatch between scenario size and question scope. Some pipelines generate elaborate multi-element scenarios but then pose only a narrow recall question, leaving the scenario largely unused. Others generate questions that could be answered without the scenario at all, making the contextual framing decorative rather than functional. The proposed pipeline consistently avoids this failure mode at medium and hard difficulty, where the scenario directly motivates the analytical or evaluative task being posed.

Baseline competitiveness. MZS+B is the most consistently strong baseline, ranking first or second in seven of eighteen settings. It is particularly competitive at easy difficulty and in settings where straightforward Bloom-conditioned generation is sufficient. ZS+B is also competitive, especially at medium difficulty, where it ranks first in the SAQ `max_score_gap` setting with a natural, well-

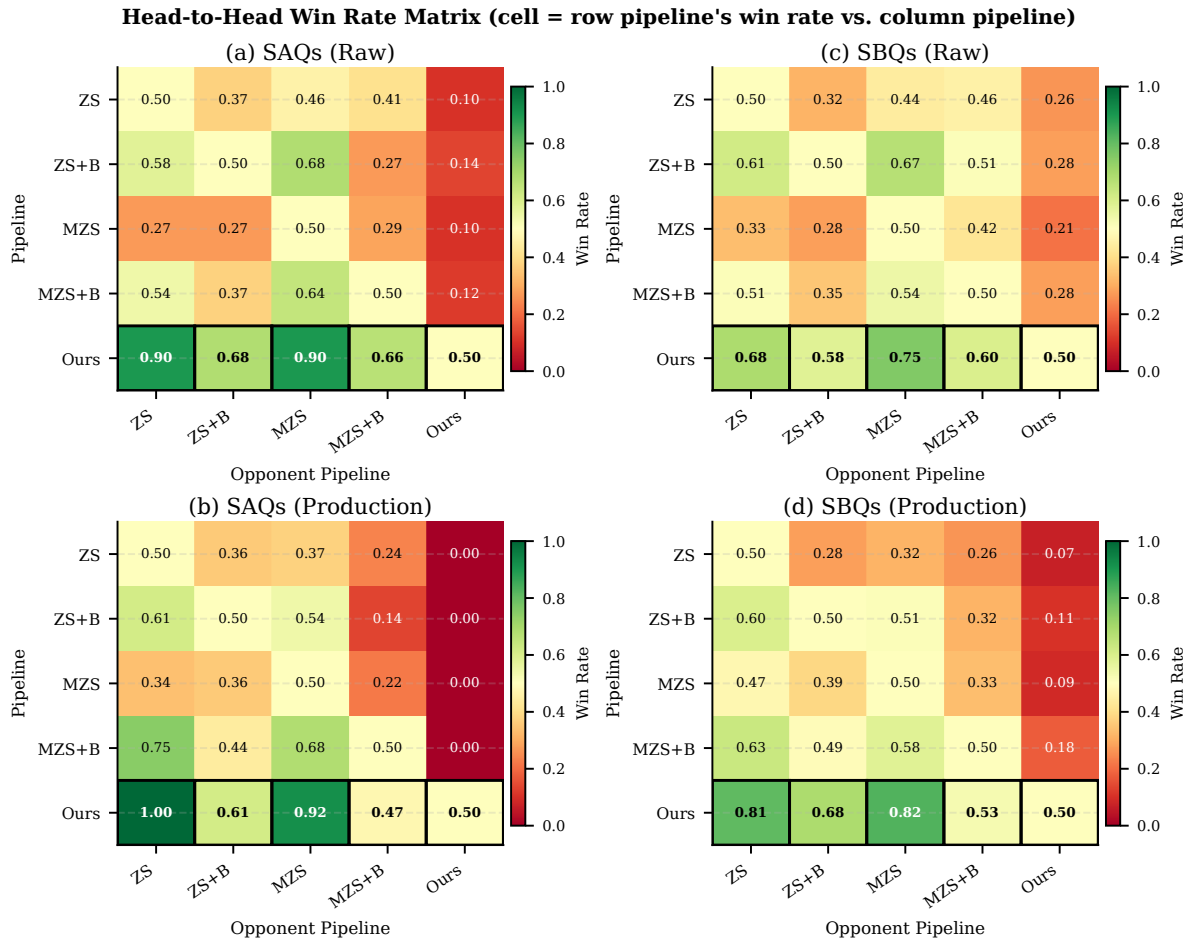

Figure 6: Head-to-head win-rate matrices for SAQs and SBQs in raw and production settings, where each cell denotes the row pipeline's win rate against the column pipeline. The diagonal is fixed at 0.50 by definition. Across all four settings, the proposed pipeline shows the strongest overall head-to-head performance.

targeted comparative Analysis question. MZS is the weakest overall baseline and never achieves a first-place finish.

Difficulty modulates the advantage. The proposed pipeline's advantage over baselines is most consistent at medium and hard difficulty. At easy difficulty, the generation task is less demanding and Bloom-aware baselines can occasionally match or exceed the proposed pipeline's output quality, particularly for SBQs. As difficulty increases and the requirement for genuine higher-order reasoning becomes the central assessment objective, the structured generation and explicit verification stages of the proposed pipeline provide a more robust and consistent advantage.

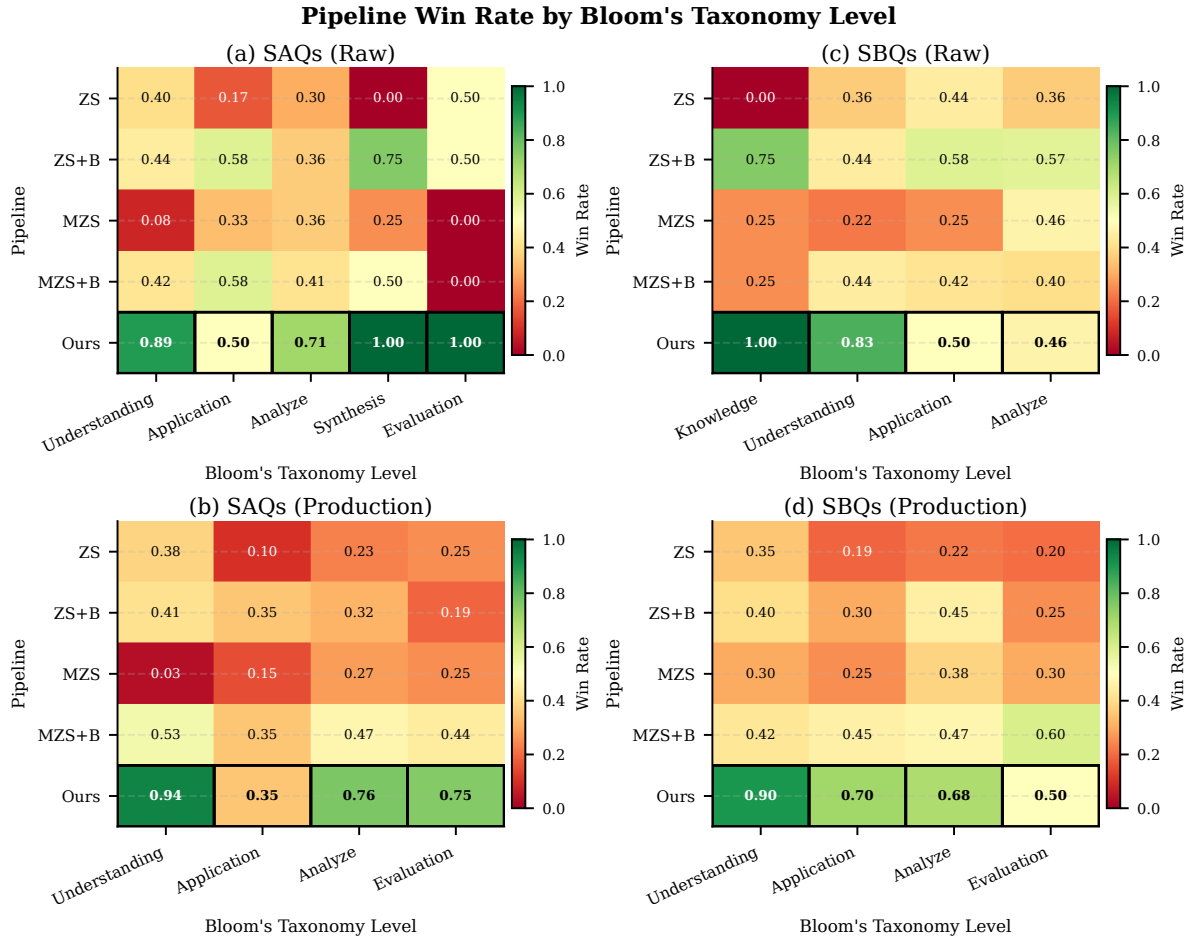

Figure 7: Win rates stratified by Bloom level for SAQs and SBQs in raw and production settings. On SAQs, the proposed pipeline generally outperforms the other pipelines across cognitive levels. On SBQs, this advantage is less uniform, with higher-order levels remaining more difficult to win consistently.

Document	Raw	Compressed	% Decrease	Category
Wk. 1 Introduction Slides	1147	984	14.21	course_adjacent
Wk. 1 Overview Text	2764	205	92.58	administrative
Wk. 2 Quiz Materials	10362	3807	63.26	course_adjacent
Wks. 2–3 Annotated Notes	**3893**	**1777**	**54.35**	**course_concept**
Wk. 2 Overview Text	2374	1011	57.41	administrative
Wk. 2 Unsupervised Learning Notes	**31866**	**5916**	**81.43**	**course_concept**
Wk. 2 Machine Learning Slides	**13350**	**4452**	**66.65**	**course_concept**
Wk. 3 Supervised Learning Notes	**26857**	**6279**	**76.62**	**course_concept**
Wk. 3 Overview Text	2009	630	68.64	administrative
Wk. 4 AI Ethics Slides	**28781**	**5580**	**80.61**	**course_concept**
Wk. 4 Ethics Case Study	**3734**	**2461**	**34.09**	**course_concept**
Wk. 4 Overview Text	1641	446	72.82	course_adjacent
Wk. 5 Overview Text	1693	279	83.52	administrative
Wk. 5 Search Slides	**29687**	**4180**	**85.92**	**course_concept**
Wk. 7 Adversarial Search Slides	**17256**	**2974**	**82.77**	**course_concept**
Wk. 7 Overview Text	**879**	**219**	**75.09**	**course_concept**
Wk. 7 Tutorial	34870	2384	93.16	course_adjacent
Wk. 7 Tutorial Solutions	**3054**	**2675**	**12.41**	**course_concept**
Wk. 8 Overview Text	1498	402	73.16	administrative
Wk. 8 CSP Slides	**23479**	**4382**	**81.34**	**course_concept**
Wk. 9 Uncertainty Slides	**20163**	**4761**	**76.39**	**course_concept**
Wk. 9 Overview Text	1811	458	74.71	administrative
Wk. 10 Lecture Slides	12138	3115	74.34	course_adjacent
Wk. 10 Overview Text	1990	453	77.24	administrative
Wk. 10 Reinforcement Learning Notes	**28052**	**5607**	**80.01**	**course_concept**
Wk. 11 Overview Text	1849	581	68.58	course_adjacent
Wk. 12 Overview Text	1006	176	82.51	administrative
Module Overview Text	4950	1033	79.13	administrative
Overall Statistics				
Total	**313153**	**67227**	**78.53**	all
Course Concept (13)	**231051**	**51263**	**77.81**	used
Non-Concept (15)	**82102**	**15964**	**80.56**	filtered

Table 11: Compression statistics for all course materials used in question generation. Columns report the original word count, compressed word count, percentage reduction, and assigned content category. Rows in bold correspond to *course_concept* items, which are retained for question generation, while administrative and adjacent content are filtered out. Overall, compression reduces total word count by 78.5% while preserving the majority of instructional content, enabling more efficient and scalable generation.

	SAQ								SBQ									
Pipeline	**OG-T**	**OG-S**	**MSQ**	**AF**	**STI**	**Cla.**	**CG**	**BLA**	**OG-T**	**OG-S**	**MSQ**	**AF**	**SRN**	**SG**	**STI**	**Cla.**	**CG**	**BLA**
ZS	0.993	0.900	0.450	0.964	0.957	0.979	0.957	0.007	0.912	0.772	0.493	0.949	0.110	0.632	0.816	0.941	0.956	0.169
ZS+B	0.979	0.915	0.366	0.972	0.993	0.986	0.965	0.718	0.984	0.814	0.605	0.961	0.171	0.589	0.884	0.961	0.907	0.643
MZS	0.908	0.839	0.238	0.946	0.916	0.977	0.962	0.073	0.959	0.728	0.500	0.967	0.098	0.553	0.882	0.935	0.931	0.142
MZS+B	0.864	0.921	0.360	0.925	0.982	0.996	0.952	0.667	0.972	0.743	0.607	0.977	0.196	0.645	0.925	0.967	0.935	0.561
Ours	**0.988**	**0.926**	**0.885**	**0.990**	**0.992**	**0.992**	**0.987**	**0.917**	0.871	0.751	**0.747**	0.885	0.137	**0.879**	**0.960**	0.887	0.852	**0.897**

Table 12: Per-criterion pass rates across pipelines for both SAQs and SBQs. SAQ criteria are OG-T = objective gradability (triple), OG-S = objective gradability (stem), MSQ = mark scheme quality, AF = answer fidelity, STI = single-task integrity, Cla. = clarity, CG = course grounding, and BLA = Bloom alignment accuracy. SBQs additionally include SRN = scenario relevance/necessity and SG = scenario grounding.

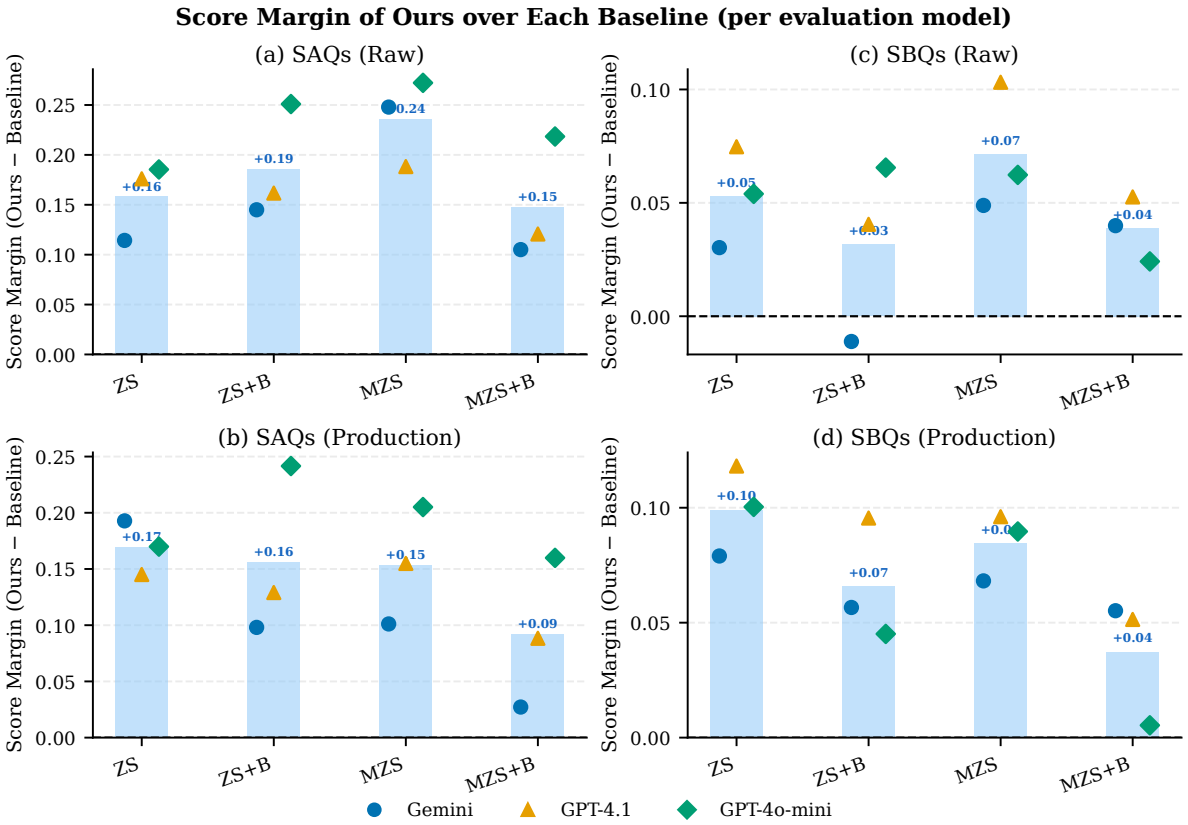

Figure 8: Mean score margin of the proposed pipeline relative to each baseline, shown separately for SAQs and SBQs in raw and production settings. Bars denote averages across evaluation models, while markers indicate per-model margins. The proposed pipeline maintains positive score margins against all baselines in every setting, with larger gains on SAQs than on SBQs.

Criterion	SAQ Weight	SBQ Weight
Mark Scheme Quality	0.326	0.160
Bloom Alignment	0.197	0.137
Obj. Gradability (Triple)	0.110	0.090
Answer Fidelity	0.093	0.089
Obj. Gradability (Stem)	0.089	0.084
Single Task Integrity	0.084	0.077
Course Grounding	0.058	0.091
Clarity	0.041	0.078
Scenario Relevance	0.000	0.070
Scenario Grounding	0.000	0.123

Pipeline	SAQ Yield	SBQ Yield
ZS	140	136
ZS+B	142	129
MZS	261	246
MZS+B	228	214
Ours	**1072**	**750**

Table 13: Left: Discrimination-based criterion weights for evaluation. Higher values indicate greater contribution to differentiating pipeline performance; scenario criteria receive zero weight for SAQs. Right: Number of valid generated assessment items (yield) per pipeline.

Illustrative Contrast SAQ Sample — Easy Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 10 — Reinforcement Learning (Lecture Notes with Solutions) | *Model:* GPT-4o-mini | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS)	Explain the significance of the single-state nature of the k-armed bandit problem compared to a Markov Decision Process (MDP) and how this difference affects the estimation of action values.	Knowledge	Analyze†	6	0.882
B (OURS)	Identify the method that selects actions uniformly at random at time t.	Knowledge	Knowledge	1	**1.000**
C (ZS)	Explain the concept of action-value estimation in k-armed bandit problems and why it is essential for decision-making.	Knowledge	Understanding†	5	0.961
D (ZS+B)	Describe how action-value estimates $Q_t(a)$ are computed in the k-armed bandit problem and explain why these estimates are necessary.	Knowledge	Understanding†	6	0.634
E (MZS+B)	Explain the significance of maintaining estimates of average rewards $Q_t(a)$ in the k-armed bandit problem and describe how these estimates are updated over time.	Understanding	Understanding	9	0.674

† Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
The Random Method selects actions uniformly at random at time t.	1
Model answer: The Random Method selects actions uniformly at random at time t.	**/1**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):* 1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
MZS *(rank 5):* 1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.4

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from easy difficulty pool to highlight a large Bloom alignment discrepancy across pipelines. **Best σ_w:** 1.000 **Bloom gap:** 0.6000 **Gap percentile rank:** 42.9$^{\text{th}}$ **Runner-up gap:** 0.6000 **Mean σ_w:** 0.8300

Blind human evaluation ranking: OURS > ZS > MZS+B > ZS+B > MZS. OURS was ranked first because it is the cleanest example of correct cognitive targeting: the declared and true Bloom levels match, the item is fully objective, and the expected answer is exact. **MZS** was ranked last because it has the largest and most consequential Bloom mismatch: it is declared as knowledge but actually asks for analytical comparison between bandits and MDPs and their implications for value estimation.

Figure 9: **Illustrative contrast SAQ sample, easy difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SAQ Sample — Medium Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2 — Introduction to Machine Learning (Slides) | *Model:* Gemini-2.5-Flash | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS+B)	Describe the core concept of machine learning by synthesizing the different definitions provided, highlighting common themes or key aspects.	Understanding	Synthesis†	8	0.679
B (OURS)	You are building a machine learning model to identify spam emails. Given a large dataset of emails, show how you would establish training and test sets to properly evaluate your model's performance on emails it has not seen before.	Application	Application	8	**1.000**
C (ZS)	Explain why accuracy alone can be an insufficient metric for evaluating machine learning models, particularly when dealing with imbalanced datasets. Provide a concise example.	Application	Understanding†	8	0.634
D (MZS)	Explain the distinction between classification and regression problems, providing an example for each to illustrate their different objectives within supervised learning.	Application	Understanding†	9	0.634
E (ZS+B)	Describe the concepts of underfitting and overfitting in machine learning, and explain how selecting the appropriate model complexity helps mitigate these issues.	Understanding	Understanding	9	0.674

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Divide the large dataset of labeled emails into two distinct sets: a training set and a test set.	2
Utilize the training set, which includes both email content and their "spam"/"not spam" labels, to teach the machine learning model to identify patterns.	1
Keep the test set separate and completely untouched during the entire training process to ensure it represents genuinely unseen data.	2
After the model is trained, use the email content from the test set to make predictions without providing the true labels to the model.	1
Compare the model's predictions on the test set against the actual (previously hidden) labels to accurately evaluate its generalization performance on new, unseen emails.	2
Model answer: To establish training and test sets, the large dataset of labeled emails is divided into two distinct sets: a training set and a test set. The training set, which includes email content and their "spam"/"not spam" labels, is utilized to teach the machine learning model to identify patterns. The test set is kept separate and completely untouched during the entire training process to ensure it represents genuinely unseen data. After the model is trained, email content from the test set is used to make predictions without providing the true labels to the model. The model's predictions on the test set are then compared against the actual (previously hidden) labels to accurately evaluate its generalization performance on new, unseen emails.	**/8**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
MZS+B *(rank 5):*							
1.0	1.0	1.0	1.0	1.0	0.5	1.0	0.2

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from medium difficulty pool to maximize a large Bloom alignment discrepancy across pipelines. **Best σ_w:** 1.000 **Bloom gap:** 0.8000 **Gap percentile rank:** 97.1th **Runner-up gap:** 0.4000 **Mean σ_w:** 0.7242

Blind human evaluation ranking: OURS > ZS+B > MZS > ZS > MZS+B. OURS was ranked first because it is the clearest example of genuine application: it uses a realistic scenario, preserves single-task integrity, and aligns perfectly with its declared Bloom level. **MZS+B** was ranked last because it has the largest Bloom mismatch in the set: the task explicitly asks students to synthesize multiple definitions of machine learning, but it is labelled only as understanding.

Figure 10: **Illustrative contrast SAQ sample, medium difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SAQ Sample — Hard Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 7 — Heuristic Search & Adversarial Search (Slides) | *Model:* GPT-4o-mini | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS)	Explain why state space complexity varies significantly among games like Tic-Tac-Toe, Chess, and Go, and discuss the implications of this variation for game search algorithms.	Evaluation	Analyze†	9	0.961
B (MZS+B)	Evaluate the conditions under which the minimax algorithm is guaranteed to be optimal and complete, and explain how these conditions relate to the nature of the opponent's play.	Evaluation	Evaluation	9	**1.000**
C (MZS)	Why is the minimax algorithm considered complete and optimal, and under what conditions might these properties fail in practical game playing?	Evaluation	Analyze†	9	0.797
D (OURS)	Evaluate whether the Minimax algorithm is a practical solution for games with extremely large state spaces such as Chess or Go, using the details about state space complexity and Minimax properties provided in the text.	Evaluation	Evaluation	9	1.000
E (ZS+B)	Discuss how Alpha-Beta pruning optimizes the Minimax algorithm and explain the roles of alpha and beta in the pruning process.	Evaluate	Understanding†	10	0.500

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Minimax explores the entire game tree with time complexity $O(b^d)$, which is infeasible for very large state spaces like Chess or Go due to enormous branching factors and depths.	2
The enormous size of Chess (8.73×10^{79} states) and Go (2×10^{170} states) makes exhaustive Minimax search practically impossible within reasonable time or space.	2
While Minimax is complete and optimal for finite trees with optimal opponents, these properties do not offset its impracticality at huge state scale.	1
Alpha-Beta pruning can reduce the number of states evaluated but still cannot make Minimax fully practical for extremely large games due to exponential growth of the tree.	2
Therefore, Minimax alone is not a practical solution for Chess or Go without significant heuristic improvements or approximations.	2
Model answer: Minimax explores the entire game tree with time complexity $O(b^d)$, which is infeasible for very large state spaces like Chess or Go due to enormous branching factors and depths. The enormous size of Chess (8.73×10^{79} states) and Go (2×10^{170} states) makes exhaustive Minimax search practically impossible within reasonable time or space. While Minimax is complete and optimal for finite trees with optimal opponents, these properties do not offset its impracticality at huge state scale. Alpha-Beta pruning can reduce the number of states evaluated but still cannot make Minimax fully practical for extremely large games due to exponential growth of the tree. Therefore, Minimax alone is not a practical solution for Chess or Go without significant heuristic improvements or approximations.	**/9**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
ZS+B *(rank 5):*							
1.0	1.0	1.0	1.0	0.5	0.0	1.0	0.4

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from hard difficulty pool to highlight a large Bloom alignment discrepancy across pipelines. **Best σ_w:** 1.000 **Bloom gap:** 0.6000 **Gap percentile rank:** 52.9$^{\text{th}}$ **Runner-up gap:** 0.6000 **Mean σ_w:** 0.8516

Blind human evaluation ranking: OURS > MZS+B > ZS > MZS > ZS+B. OURS was ranked first because it is the most naturally evaluative item and aligns perfectly with its declared Bloom level: it asks for a justified practicality judgment under explicit state-space constraints. **ZS+B** was ranked last because it is the clearest case of a question labeled as evaluation while actually requiring only understanding of alpha-beta pruning and the roles of alpha and beta.

Figure 11: **Illustrative contrast SAQ sample, hard difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SAQ Sample — Easy Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 8 — Planning & CSPs (Lecture Slides) | *Model:* GPT-4.1 | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS)	Explain the three main components of a Constraint Satisfaction Problem (CSP) and describe how these are represented in the map colouring and n-Queens examples.	Knowledge	Understanding†	6	0.634
B (ZS+B)	Describe the structure of a Constraint Satisfaction Problem (CSP), using the map colouring example to illustrate your explanation.	Understanding	Understanding	6	0.674
C (MZS+B)	Explain the three main components that define a Constraint Satisfaction Problem (CSP), and illustrate how each component is specified in the example of map colouring for Australia.	Understanding	Understanding	10	0.674
D (OURS)	Define a Constraint Satisfaction Problem (CSP).	Knowledge	Knowledge	3	**1.000**
E (MZS)	Explain the purpose of heuristics such as the Minimum Remaining Values (MRV) and Least Constraining Value in the context of CSP solving.	Knowledge	Understanding†	8	0.590

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
A CSP consists of a finite set of variables.	1
Each variable has a domain of possible values.	1
A set of constraints specifies allowable combinations of variable values.	1
Model answer: A CSP consists of a finite set of variables, each variable has a domain of possible values, and a set of constraints specifies allowable combinations of variable values.	**/3**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
MZS *(rank 5):*							
1.0	1.0	1.0	0.5	1.0	0.0	1.0	0.8

Selection Justification

Automated selection (max score gap axis): task selected from easy difficulty pool to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best σ_w:** 1.000 **Worst σ_w:** 0.5895 **Score gap:** 0.4105 **Gap percentile rank:** 85.7th **Runner-up gap:** 0.3658 **Mean σ_w:** 0.7142

Blind human evaluation ranking: OURS > ZS+B > MZS+B > ZS > MZS. OURS was ranked first because it is the most internally consistent triple: exact Bloom alignment, highly objective grading, clear wording, and full answer-to-scheme correspondence. **MZS** was ranked last because it is Bloom-misaligned and less tightly constrained as an easy item: asking for the *purpose* of MRV and Least Constraining Value requires explanation rather than simple recall.

Figure 12: **Illustrative contrast SAQ sample, easy difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SAQ Sample — Medium Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 7 — Heuristic Search & Adversarial Search (Text) | *Model:* GPT-4.1 | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (OURS)	Apply the concept of heuristics to inform the strategy for an adversarial search in a board game situation.	Application	Application	6	0.900
B (ZS+B)	Compare ordinary search and adversarial search, highlighting the key differences between the two approaches.	Analyze	Analyze	8	0.833
C (ZS)	Explain how relaxations are used to derive heuristics in search, and describe why heuristics are valuable in guiding the search process.	Application	Understanding†	7	0.425
D (MZS)	Explain how relaxations can be used to derive heuristics in the context of guiding search algorithms.	Application	Understanding†	9	0.447
E (MZS+B)	Explain how relaxations can be used to derive heuristics for guiding search processes.	Application	Understanding†	9	0.467

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (ZS+B — rank 1):

Marking point	Marks
Ordinary search involves finding a solution without opposition.	1
Adversarial search considers the presence of an opponent making counter-moves.	2
Key difference lies in the need to anticipate and respond to adversary strategies in adversarial search.	2
Ordinary search focuses solely on optimization or reaching a goal.	1
Adversarial search requires the evaluation of possible outcomes based on both parties' actions.	2
Model answer: Ordinary search focuses on finding a solution or optimizing a path to reach a goal, without any opposition. In contrast, adversarial search involves anticipating and responding to an opponent's actions, requiring strategies that account for counter-moves. The key difference is that adversarial search must evaluate possible outcomes based on the choices of both parties, while ordinary search simply optimizes or searches for a goal without considering adversarial interactions.	**/8**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	BLA
ZS+B *(rank 1):* 0.5	1.0	1.0	0.5	1.0	**1.0**	0.0	1.0
ZS *(rank 5):* 0.5	1.0	0.5	0.5	1.0	0.0	0.0	0.8

Selection Justification

Automated selection (max score gap axis): task selected from medium difficulty pool to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best σ_w:** 0.9001 **Worst σ_w:** 0.4253 **Score gap:** 0.4748 **Gap percentile rank:** 97.1$^{\text{th}}$ **Runner-up gap:** 0.4572 **Mean σ_w:** 0.6145

Blind human evaluation ranking: ZS+B > OURS > MZS+B > MZS > ZS. ZS+B was ranked first because it offers the most natural and properly targeted medium-level task: a clear comparative analysis question with good course grounding and a coherent mark scheme. **ZS** was ranked last because it is both Bloom-misaligned and less focused as a single assessment objective, combining explanation of relaxations with a broader discussion of why heuristics are useful.

Figure 13: **Illustrative contrast SAQ sample, medium difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SAQ Sample — Hard Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 7 — Heuristic Search & Adversarial Search (Slides) | *Model:* GPT-4o-mini | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS+B)	Discuss how Alpha-Beta pruning optimizes the Minimax algorithm and explain the roles of alpha and beta in the pruning process.	Evaluation	Understanding†	10	0.500
B (OURS)	Evaluate whether the Minimax algorithm is a practical solution for games with extremely large state spaces such as Chess or Go, using the details about state space complexity and Minimax properties provided in the text.	Evaluation	Evaluation	9	**1.000**
C (MZS)	Why is the minimax algorithm considered complete and optimal, and under what conditions might these properties fail in practical game playing?	Evaluation	Analyze†	9	0.797
D (MZS+B)	Evaluate the conditions under which the minimax algorithm is guaranteed to be optimal and complete, and explain how these conditions relate to the nature of the opponent's play.	Evaluation	Evaluation	9	1.000
E (ZS)	Explain why state space complexity varies significantly among games like Tic-Tac-Toe, Chess, and Go, and discuss the implications of this variation for game search algorithms.	Evaluation	Analyze†	9	0.961

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Minimax explores the entire game tree with time complexity $O(b^d)$, which is infeasible for very large state spaces like Chess or Go due to enormous branching factors and depths.	2
The enormous size of Chess (8.73×10^{79} states) and Go (2×10^{170} states) makes exhaustive Minimax search practically impossible within reasonable time or space.	2
While Minimax is complete and optimal for finite trees with optimal opponents, these properties do not offset its impracticality at huge state scale.	1
Alpha-Beta pruning can reduce the number of states evaluated but still cannot make Minimax fully practical for extremely large games due to exponential growth of the tree.	2
Therefore, Minimax alone is not a practical solution for Chess or Go without significant heuristic improvements or approximations.	2
Model answer: Minimax explores the entire game tree with time complexity $O(b^d)$, which is infeasible for very large state spaces like Chess or Go due to enormous branching factors and depths. The enormous size of Chess (8.73×10^{79} states) and Go (2×10^{170} states) makes exhaustive Minimax search practically impossible within reasonable time or space. While Minimax is complete and optimal for finite trees with optimal opponents, these properties do not offset its impracticality at huge state scale. Alpha-Beta pruning can reduce the number of states evaluated but still cannot make Minimax fully practical for extremely large games due to exponential growth of the tree. Therefore, Minimax alone is not a practical solution for Chess or Go without significant heuristic improvements or approximations.	**/9**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):* 1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
ZS+B *(rank 5):* 1.0	1.0	1.0	1.0	0.5	0.0	1.0	0.4

Selection Justification

Automated selection (max score gap axis): task selected from hard difficulty pool to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best σ_w:** 1.000 **Worst σ_w:** 0.5000 **Score gap:** 0.5000 **Gap percentile rank:** 94.1th **Runner-up gap:** 0.4448 **Mean σ_w:** 0.8516

Blind human evaluation ranking: OURS > MZS+B > ZS > MZS > ZS+B. OURS was ranked first because it is the strongest example of a true hard-level evaluative question: it requires a clear judgment on practicality, is strongly grounded in course content, and has an excellent mark-scheme structure. **ZS+B** was ranked last because it is fundamentally an explanatory understanding task mislabeled as evaluation: the mark scheme rewards description of alpha-beta pruning rather than genuine critique or judgment.

Figure 14: **Illustrative contrast SAQ sample, hard difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Easy Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2–3 — Introduction to Machine Learning & Unsupervised Learning (Annotated Notes) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS)	If the algorithm always predicts "no disease," calculate its accuracy, and discuss whether this is a good way to judge the algorithm's performance in this scenario.	Knowledge	Analyze†	5	0.685
B (ZS+B)	Define the metrics MAE, MSE, and RMSE in the context of regression, and explain how each metric measures prediction errors.	Understanding	Understanding	5	0.965
C (ZS)	Which terms describe these two parts of the dataset, and why is it important to keep the evaluation set labels hidden during training?	Knowledge	Analyze†	4	0.685
D (OURS)	Identify the two data splits used for model development in the scenario.	Knowledge	Knowledge	2	0.965
E (MZS+B)	Describe the roles of the training set and the test set in supervised learning.	Knowledge	Knowledge	**3**	**0.965**

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (MZS+B — rank 1):

Marking point	Marks
The training set is used to fit or train the model by providing both input data and their corresponding labels.	1
The test set is used to evaluate the trained model's performance by using input data with hidden labels during prediction.	1
The test set helps assess how well the model generalizes to unseen data.	1
Model answer: In supervised learning, the training set serves to train the model by providing input data along with their corresponding labels, allowing the model to learn the relationship between features and outcomes. After training, the test set is used to evaluate the model's performance by making predictions on input data for which the labels are hidden; this process assesses how well the model generalizes to new, unseen data.	**/3**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
MZS+B *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS *(rank 5):*									
1.0	1.0	0.5	1.0	1.0	0.0	1.0	0.5	1.0	0.4

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from easy difficulty pool (n=11 candidates) to highlight a large gap in Bloom alignment quality under otherwise matched generation conditions. **Best σ_w:** 0.9648 **Bloom gap:** 0.6000 **Gap percentile rank:** 72.7$^{\text{th}}$ **Runner-up gap:** 0.6000 **Mean σ_w:** 0.8527

Blind human evaluation ranking: MZS+B > ZS+B > OURS > ZS > MZS. MZS+B was ranked first because it most cleanly matches its declared knowledge-level target while remaining clear, grounded, and objective; it asks for the roles of the training and test sets and uses the scenario proportionately. **MZS** was ranked last because it has the clearest Bloom mismatch in the set: it is labelled as knowledge, but calculating accuracy and judging its appropriateness in an imbalanced setting requires analysis.

Figure 15: **Illustrative contrast SBQ sample, easy difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Medium Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2 — Introduction to Machine Learning (Slides) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS)	Explain what underfitting and overfitting mean in the context of this scenario, and identify which degrees correspond to each phenomenon.	Application	Understanding†	6	0.778
B (MZS)	Explain why the algorithm's high accuracy does not necessarily indicate it is a reliable or effective diagnostic tool in this context.	Application	Understanding†	7	0.716
C (MZS+B)	Explain why the algorithm's high accuracy does not necessarily indicate good performance in this context, and suggest a more appropriate evaluation metric.	Understanding	Evaluation†	6	0.779
D (OURS)	Compare the training and validation MSE values for polynomial degrees 1, 4, and 29, and analyze the pattern that emerges as the model complexity increases.	Analyze	Analyze	**6**	**0.965**
E (ZS+B)	Using the concepts of underfitting and overfitting, explain why the models of degree 1 and 29 perform poorly on validation data, and why the degree 4 model is preferable.	Application	Understanding†	6	0.778

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
For degree 1, training MSE is 0.189 and validation MSE is 0.379, indicating moderate fit.	1
For degree 4, training MSE drops to 0.00691 and validation MSE to 0.0120, showing improved fit and generalization.	2
For degree 29, training MSE is extremely low at 0.00322, but validation MSE spikes to 1.76×10^{15}, signaling severe overfitting.	2
As model complexity increases, training error decreases steadily but validation error decreases then dramatically increases, revealing the overfitting pattern.	1
Model answer: For degree 1, training MSE is 0.189 and validation MSE is 0.379, indicating moderate fit. For degree 4, training MSE drops to 0.00691 and validation MSE to 0.0120, showing improved fit and generalization. For degree 29, training MSE is extremely low at 0.00322, but validation MSE spikes to 1.76×10^{15}, signaling severe overfitting. As model complexity increases, training error decreases steadily but validation error decreases then dramatically increases, revealing the overfitting pattern.	**/6**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS+B *(rank 5):*									
1.0	1.0	1.0	0.5	1.0	1.0	1.0	0.5	0.5	0.4

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from medium difficulty pool to maximize a large Bloom alignment discrepancy across pipelines. **Best σ_w:** 0.9648 **Bloom gap:** 0.6000 **Gap percentile rank:** 96.7$^{\text{th}}$ **Runner-up gap:** 0.4000 **Mean σ_w:** 0.8032

Blind human evaluation ranking: OURS > ZS+B > ZS > MZS > MZS+B. OURS was ranked first because it most cleanly matches its declared analyze-level target and uses the scenario values directly to support genuine comparison and trend interpretation. **MZS+B** was ranked last because it is declared as understanding but actually asks for a more evaluative judgment by requiring students to suggest a better metric for the scenario.

Figure 16: **Illustrative contrast SBQ sample, medium difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Hard Difficulty (Bloom Alignment Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 9 — MDPs / Uncertainty (Slides) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS+B)	Design a suitable reward function for the robot's navigation MDP. Briefly justify your choices for how rewards should be assigned to regular, goal, and hazardous states.	Synthesis	Synthesis	5	0.702
B (OURS)	Assess the effectiveness of modeling the warehouse inventory management system as a Markov Decision Process, given the described state space, actions, and transition uncertainties.	Evaluation	Evaluation	**8**	**0.965**
C (MZS+B)	Discuss how the reward function influences the policy derived from value iteration in such an uncertain inventory system. What might be the consequences of missing or poorly designed reward components?	Evaluation	Analyze†	8	0.938
D (MZS)	Explain the process for updating the utility value of state $(2, 1)$ using the Bellman equation and discuss what the value 0.655 represents in this context.	Evaluation	Understanding†	8	0.723
E (ZS)	Apply the Bellman equation to compute the updated utility $U((2, 1))$ for moving left, showing your calculation, and determine which action yields the maximum expected utility.	Evaluation	Application†	6	0.910

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
MDP formalism allows systematic modeling of sequential decisions, capturing inventory dynamics through defined states and actions.	1
Probabilistic state transitions in the MDP match real-world uncertainties such as customer demand variability and delivery timing.	2
The MDP reward structure effectively incorporates holding costs, ordering costs, and expected profits, supporting quantitative evaluation of policies.	1
The large state space may increase computational complexity, potentially limiting tractability for optimal policy computation.	2
MDP modeling is suitable for inventory management when transition probabilities and rewards are well-defined, but performance hinges on accurate probability estimation and scalable solution methods.	2
Model answer: MDP formalism allows systematic modeling of sequential decisions, capturing inventory dynamics through defined states and actions. Probabilistic state transitions in the MDP match real-world uncertainties such as customer demand variability and delivery timing. The MDP reward structure effectively incorporates holding costs, ordering costs, and expected profits, supporting quantitative evaluation of policies. The large state space may increase computational complexity, potentially limiting tractability for optimal policy computation. MDP modeling is suitable for inventory management when transition probabilities and rewards are well-defined, but performance hinges on accurate probability estimation and scalable solution methods.	**/8**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS *(rank 5):*									
1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.5	1.0	0.4

Selection Justification

Automated selection (Bloom alignment gap axis): task selected from hard difficulty pool to highlight a large gap in Bloom alignment quality under otherwise matched generation conditions. **Best σ_w:** 0.9648 **Bloom gap:** 0.6000 **Gap percentile rank:** 81.2$^{\text{th}}$ **Runner-up gap:** 0.6000 **Mean σ_w:** 0.8474

Blind human evaluation ranking: OURS > ZS+B > MZS+B > ZS > MZS. OURS was ranked first because it is the clearest example of exact Bloom targeting in the set: the task is genuinely evaluative, the scenario is necessary, and the mark scheme supports a justified conclusion on whether the MDP formulation is suitable. **MZS** was ranked last because it has the largest and clearest Bloom mismatch: explaining a Bellman update and interpreting a value is an understanding task, not an evaluation task.

Figure 17: **Illustrative contrast SBQ sample, hard difficulty tier (Bloom alignment gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Easy Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2–3 — Introduction to Machine Learning & Unsupervised Learning (Annotated Notes) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS)	If the algorithm always predicts "no disease," calculate its accuracy, and discuss whether this is a good way to judge the algorithm's performance in this scenario.	Knowledge	Analyze†	5	0.685
B (ZS)	Which terms describe these two parts of the dataset, and why is it important to keep the evaluation set labels hidden during training?	Knowledge	Analyze†	4	0.685
C (MZS+B)	Describe the roles of the training set and the test set in supervised learning.	Knowledge	Knowledge	3	**0.965**
D (ZS+B)	Define the metrics MAE, MSE, and RMSE in the context of regression, and explain how each metric measures prediction errors.	Understanding	Understanding	5	0.965
E (OURS)	Identify the two data splits used for model development in the scenario.	Knowledge	Knowledge	2	0.965

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (MZS+B — rank 1):

Marking point	Marks
The training set is used to fit or train the model by providing both input data and their corresponding labels.	1
The test set is used to evaluate the trained model's performance by using input data with hidden labels during prediction.	1
The test set helps assess how well the model generalizes to unseen data.	1
Model answer: In supervised learning, the training set serves to train the model by providing input data along with their corresponding labels, allowing the model to learn the relationship between features and outcomes. After training, the test set is used to evaluate the model's performance by making predictions on input data for which the labels are hidden; this process assesses how well the model generalizes to new, unseen data.	**/3**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
MZS+B *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS *(rank 5):*									
1.0	1.0	0.5	1.0	1.0	0.0	1.0	0.5	1.0	0.4

Selection Justification

Automated selection (max score gap axis): task selected from easy difficulty pool (n=11 candidates) to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best σ_w:** 0.9648 **Worst σ_w:** 0.6845 **Score gap:** 0.2803 **Gap percentile rank:** 90.9$^{\text{th}}$ **Runner-up gap:** 0.2416 **Mean σ_w:** 0.8527

Blind human evaluation ranking: MZS+B > ZS+B > OURS > ZS > MZS. MZS+B was ranked first because it is the cleanest and most internally consistent SBQ in the set: the question is a straightforward knowledge-level task, the declared and true Bloom levels match exactly, and the mark scheme is concise, objective, and fully aligned with the expected answer. **MZS** was ranked last because it combines numerical calculation with critique of metric suitability in an imbalanced classification setting, making it analytically demanding despite being labelled as knowledge.

Figure 18: **Illustrative contrast SBQ sample, easy difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Medium Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 8 — Planning & CSPs (Lecture Slides) | *Model:* Gemini-2.5-Flash | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS+B)	Describe how Forward Checking would update the domains of variables B and C immediately after A is assigned "Red".	Analyze	Understanding†	6	0.647
B (ZS+B)	Given that DevA is assigned to Task1, and knowing the "all different" constraint applies to both developers and tasks, apply the Minimum Remaining Values (MRV) heuristic to determine which unassigned developer should be selected next. Show the updated legal domains and explain your choice.	Application	Application	6	0.903
C (MZS)	Using the Most Constrained Variable (MRV) heuristic, which variable should the agent select next for assignment, and why?	Application	Application	4	0.903
D (OURS)	Compute the total number of initial, unconstrained colour assignments possible for all 7 regions of the map using the given colour domain.	Application	Application	**7**	**0.965**
E (ZS)	Explain two specific heuristics, one for variable selection and one for value ordering, that the company could implement to significantly improve the efficiency of their backtracking search.	Application	Application	4	0.965

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Identifies the total number of regions (variables) on the map as 7.	1
Identifies the size of the colour domain (values) as 3 (red, green, blue).	1
Applies the combinatorial principle for independent assignments, using the number of colours as the base and the number of regions as the exponent (3^7).	2
Calculates the result of 3^7 correctly.	2
States the final computed total number of initial unconstrained assignments as 2187.	1
Model answer: With 7 regions on the map and a colour domain size of 3 (red, green, blue), the total number of initial unconstrained assignments is calculated as 3^7, which equals 2187.	**/7**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS+B *(rank 5):*									
1.0	1.0	1.0	0.5	1.0	0.0	1.0	0.5	0.5	0.6

Selection Justification

Automated selection (max score gap axis): task selected from medium difficulty pool to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best** σ_w**:** 0.9648 **Worst** σ_w**:** 0.6469 **Score gap:** 0.3179 **Gap percentile rank:** 96.7th **Runner-up gap:** 0.2484 **Mean** σ_w**:** 0.8767

Blind human evaluation ranking: OURS > ZS > ZS+B > MZS > MZS+B. OURS was ranked first because it is the most internally consistent and objectively gradable SBQ in the set: the task is a sharply defined application of combinatorial reasoning, the declared and true Bloom levels match, and the answer maps directly to a logically sequenced mark scheme. **MZS+B** was ranked last because it overstates the cognitive level: describing Forward Checking domain updates after a fixed assignment is procedural understanding rather than genuine analysis.

Figure 19: **Illustrative contrast SBQ sample, medium difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Illustrative Contrast SBQ Sample — Hard Difficulty (Max Score Gap Axis)

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 9 — MDPs / Uncertainty (Slides) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS+B)	Design a suitable reward function for the robot's navigation MDP. Briefly justify your choices for how rewards should be assigned to regular, goal, and hazardous states.	Synthesis	Synthesis	5	0.702
B (ZS)	Apply the Bellman equation to compute the updated utility $U((2,1))$ for moving left, showing your calculation, and determine which action yields the maximum expected utility.	Evaluation	Application†	6	0.910
C (OURS)	Assess the effectiveness of modeling the warehouse inventory management system as a Markov Decision Process, given the described state space, actions, and transition uncertainties.	Evaluation	Evaluation	**8**	**0.965**
D (MZS)	Explain the process for updating the utility value of state $(2,1)$ using the Bellman equation and discuss what the value 0.655 represents in this context.	Evaluation	Understanding†	8	0.723
E (MZS+B)	Discuss how the reward function influences the policy derived from value iteration in such an uncertain inventory system. What might be the consequences of missing or poorly designed reward components?	Evaluation	Analyze†	8	0.938

† Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
MDP formalism allows systematic modeling of sequential decisions, capturing inventory dynamics through defined states and actions.	1
Probabilistic state transitions in the MDP match real-world uncertainties such as customer demand variability and delivery timing.	2
The MDP reward structure effectively incorporates holding costs, ordering costs, and expected profits, supporting quantitative evaluation of policies.	1
The large state space may increase computational complexity, potentially limiting tractability for optimal policy computation.	2
MDP modeling is suitable for inventory management when transition probabilities and rewards are well-defined, but performance hinges on accurate probability estimation and scalable solution methods.	2
Model answer: MDP formalism allows systematic modeling of sequential decisions, capturing inventory dynamics through defined states and actions. Probabilistic state transitions in the MDP match real-world uncertainties such as customer demand variability and delivery timing. The MDP reward structure effectively incorporates holding costs, ordering costs, and expected profits, supporting quantitative evaluation of policies. The large state space may increase computational complexity, potentially limiting tractability for optimal policy computation. MDP modeling is suitable for inventory management when transition probabilities and rewards are well-defined, but performance hinges on accurate probability estimation and scalable solution methods.	**/8**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS *(rank 5):*									
1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.5	1.0	0.4

Selection Justification

Automated selection (max score gap axis): task selected from hard difficulty pool to maximise the spread between the highest- and lowest-scoring pipelines on the discrimination-weighted criterion score σ_w. **Best σ_w:** 0.9648 **Worst σ_w:** 0.7015 **Score gap:** 0.2633 **Gap percentile rank:** 93.8$^{\text{th}}$ **Runner-up gap:** 0.2565 **Mean σ_w:** 0.8474

Blind human evaluation ranking: OURS > MZS+B > ZS > ZS+B > MZS. OURS was ranked first because it is the clearest example of a genuine evaluation-level SBQ in the set: the scenario is essential, the question requires a justified judgment about MDP suitability, and the mark scheme balances strengths and limitations of the formulation. **MZS** was ranked last because it substantially inflates Bloom level: explaining a Bellman update and interpreting the resulting value is a conceptual understanding task, not an evaluation task.

Figure 20: **Illustrative contrast SBQ sample, hard difficulty tier (max score gap axis).** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Representative SAQ Sample — Easy Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 7 — Heuristic Search & Adversarial Search (Slides) | *Model:* GPT-4.1 | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS+B)	Explain the role and structure of game trees in modeling two-player adversarial games, including the significance of MAX, MIN, and terminal nodes.	Understanding	Understanding	9	0.674
B (ZS+B)	Explain the structure and purpose of a game tree in the context of two-player adversarial games, naming each type of node and describing their relationships.	Understanding	Understanding	7	0.674
C (OURS)	Define what a game tree is in the context of two-player games.	Knowledge	Knowledge	3	**1.000**
D (ZS)	Explain the structure of a game tree in two-player adversarial games, describing the roles of MAX, MIN, and terminal nodes.	Knowledge	Understanding†	5	0.961
E (MZS)	Explain the roles of MAX and MIN players in a two-player adversarial game and how game trees are used to represent the sequence of possible moves.	Knowledge	Understanding†	7	0.634

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
A game tree is a tree structure where nodes represent game states and edges represent moves.	1
It models two-player games with alternating MAX and MIN player turns.	1
Terminal nodes represent final outcomes of the game and have no children.	1
Model answer: A game tree is a tree structure where nodes represent game states and edges represent moves, it models two-player games with alternating MAX and MIN player turns, and terminal nodes represent final outcomes of the game and have no children.	**/3**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	$\text{Gradability}_{\text{stem}}$	$\text{Gradability}_{\text{triple}}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
MZS *(rank 5):*							
1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.8

Selection Justification

Representative matched-task selection: task selected from the easy difficulty pool (n=7 candidates; selected percentile = 42.9) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.7884

Blind human evaluation ranking: OURS > ZS+B > MZS+B > ZS > MZS. OURS was ranked first because it is the most precise, controlled, and internally consistent assessment triple: the question is tightly scoped, the declared and true Bloom levels match exactly, and the mark scheme and expected answer align perfectly around a single knowledge-level objective. **MZS** was ranked last because, although grounded and answerable, it is less tightly structured and Bloom-misaligned: it asks for conceptual explanation of MAX/MIN roles and game-tree representation while being labelled only as knowledge.

Figure 21: **Representative SAQ sample, easy difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Representative SAQ Sample — Medium Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 7 — Heuristic Search & Adversarial Search (Answers) | *Model:* GPT-4.1 | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (OURS)	Apply the concept of admissible heuristics to select an appropriate heuristic for an A* search in a new maze, ensuring the optimal path is found.	Application	Application	5	**1.000**
B (MZS+B)	Compare the effects of scaling an admissible heuristic by a factor greater than 1 on Greedy Best-First Search versus A* search. Why do these effects differ?	Analyze	Analyze	10	0.782
C (ZS)	Describe what is meant by an admissible heuristic and discuss how admissibility and informativeness affect the performance and guarantees of A* search.	Application	Analyze†	8	0.797
D (MZS)	Explain the differences in the paths chosen by Greedy Best-First Search and A* Search in the given example, including the reasons behind their respective path costs.	Application	Analyze†	9	0.797
E (ZS+B)	Explain the difference between Greedy Best-First Search (GBFS) and A* Search in terms of the paths and path costs they find in the given example.	Understanding	Understanding	8	0.674

†Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Choose a heuristic that never over-estimates the true cost from any node to the goal, satisfying admissibility.	1
Verify the selected heuristic is admissible so A* will guarantee finding the optimal path through the maze.	2
Avoid using scaled heuristics (such as multiplying by a constant greater than 1) since this may cause over-estimation and inadmissibility.	1
Consider using a commonly admissible heuristic like Manhattan distance for maze environments, as it estimates the minimum number of steps to the goal without overestimating.	1
Model answer: Select a heuristic that never over-estimates the true cost from any node to the goal to ensure admissibility, verify this property to guarantee that A* finds the optimal path, do not use scaled heuristics with constants greater than 1 as they may over-estimate, and use an admissible heuristic like Manhattan distance for a maze since it does not overestimate the number of steps to the goal.	**/5**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	BLA
OURS *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
ZS *(rank 5):*							
1.0	1.0	1.0	1.0	1.0	0.5	1.0	0.8

Selection Justification

Representative matched-task selection: task selected from the medium difficulty pool (n=35 candidates; selected percentile = 48.6) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.8100

Blind human evaluation ranking: OURS > MZS+B > ZS+B > MZS > ZS. OURS was ranked first because it is the most complete and internally consistent assessment triple: it poses a genuine application-level task, operationalizes admissibility concretely, and supports highly objective grading through a tightly matched mark scheme and answer. **ZS** was ranked last because it is Bloom-misaligned and too abstract for its declared task type: it frames a broad conceptual discussion of admissibility and informativeness rather than a concrete application to a specific case.

Figure 22: **Representative SAQ sample, medium difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Representative SAQ Sample — Hard Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2 — Introduction to Machine Learning & Unsupervised Learning (Lecture Notes with Solutions) | *Model:* GPT-4o-mini | *Task type:* Short Answer Question (SAQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS)	Compare and contrast the Single-Linkage, Complete-Linkage, and Average-Linkage criteria in hierarchical clustering, focusing on how they define the distance between clusters.	Evaluation	Analyze†	13	0.634
B (MZS+B)	Given a practical scenario where delivery by helicopter is limited to a 100-mile range, justify why Complete-Linkage clustering would be preferable over other linkage criteria for forming suitable clusters.	Evaluation	Evaluation	10	**1.000**
C (ZS+B)	Compare and contrast the single-linkage, complete-linkage, and average-linkage criteria used in hierarchical clustering to measure cluster distances.	Analyze	Analyze	10	1.000
D (OURS)	Assess the effectiveness of running K-means multiple times with different initialisations or using k-means++ to avoid local minima, as described in the text.	Evaluation	Evaluation	8	1.000
E (MZS)	Given a practical scenario where a helicopter delivery service must cluster cities such that any city in a cluster is reachable within a 100-mile flight, discuss why the Complete-Linkage criterion is the suitable choice and how it ensures this constraint.	Evaluation	Application†	10	0.595

† Declared–true Bloom mismatch identified by blind human evaluator.

Mark scheme and expected answer (MZS+B — rank 1):

Marking point	Marks
Complete-Linkage clustering forms clusters based on the maximum distance between points, ensuring all points in a cluster are within a certain range.	2
This method tends to produce compact, tight clusters, which aligns with the helicopter's 100-mile delivery range constraint.	2
In contrast, Single-Linkage clustering can create elongated clusters where some points are far apart, possibly exceeding the 100-mile limit.	1
Average-Linkage clustering averages distances, which may allow some points in a cluster to be beyond the 100-mile range even if the average distance is acceptable.	1
Complete-Linkage minimizes the risk of any delivery exceeding the 100-mile range, which is critical for operational feasibility and safety.	2
Using Complete-Linkage supports clear decision-making by providing well-defined cluster boundaries that respect the practical distance limit.	1
Overall, Complete-Linkage clustering better guarantees deliveries stay within the helicopter's operational range compared to other linkage methods.	1
Model answer: Complete-Linkage clustering forms clusters based on the maximum distance between points, ensuring all points within a cluster remain within the helicopter's 100-mile delivery range. This method produces compact, tight clusters, aligning closely with the operational constraint, unlike Single-Linkage clustering which can generate elongated clusters with points exceeding the range. Average-Linkage, by averaging distances, may also include points beyond the limit despite an acceptable average. Therefore, Complete-Linkage minimizes the risk of any delivery exceeding the range, supports clear decision-making through well-defined cluster boundaries, and overall better guarantees feasible and safe helicopter deliveries.	**/10**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{\text{stem}}$	Gradability$_{\text{triple}}$	Mk. scheme	Fidelity	BLA
MZS+B *(rank 1):*							
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0
MZS *(rank 5):*							
1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.6

Selection Justification

Representative matched-task selection: task selected from the hard difficulty pool (n=17 candidates; selected percentile = 47.1) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.8458

Blind human evaluation ranking: MZS+B > OURS > ZS+B > ZS > MZS. MZS+B was ranked first because it is a clear, well-grounded evaluation task embedded in a realistic constraint-driven scenario: it requires comparison of linkage criteria, judgment under the 100-mile range constraint, and an explicitly justified decision. **MZS** was ranked last because it underdelivers on evaluation: although grounded in a plausible scenario, it is closer to application than genuine evaluative reasoning and does not sufficiently require comparison across methods under competing considerations.

Figure 23: **Representative SAQ sample, hard difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Representative SBQ Sample — Easy Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2–3 — Introduction to Machine Learning & Unsupervised Learning (Annotated Notes) | *Model:* GPT-4.1 | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Easy

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (OURS)	Identify the two data splits used for model development in the scenario.	Knowledge	Knowledge	2	0.965
B (ZS+B)	Define the metrics MAE, MSE, and RMSE in the context of regression, and explain how each metric measures prediction errors.	Understanding	Understanding	5	**0.965**
C (MZS+B)	Describe the roles of the training set and the test set in supervised learning.	Knowledge	Knowledge	3	0.965
D (ZS)	Which terms describe these two parts of the dataset, and why is it important to keep the evaluation set labels hidden during training?	Knowledge	Analyze†	4	0.685
E (MZS)	If the algorithm always predicts "no disease," calculate its accuracy, and discuss whether this is a good way to judge the algorithm's performance in this scenario.	Knowledge	Analyze†	5	0.685

†Declared–true Bloom mismatch identified by blind human evaluator.

Scenario (ZS+B — rank 1):

> A data scientist builds a regression model to predict housing prices based on several features. After training the model, she evaluates it on a test set and calculates the Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root-Mean-Squared Error (RMSE).

Mark scheme and expected answer (ZS+B — rank 1):

Marking point	Marks
Defines Mean Absolute Error (MAE) as the average absolute difference between predicted and true values.	1
Defines Mean Squared Error (MSE) as the average squared difference between predicted and true values.	1
Defines Root-Mean-Squared Error (RMSE) as the square root of the average squared difference.	1
Explains that MAE measures average magnitude of errors, MSE penalizes larger errors, and RMSE brings scale back to original units.	2
Model answer: Mean Absolute Error (MAE) is calculated as the average absolute difference between the predicted and actual housing prices. Mean Squared Error (MSE) measures the average of the squared differences, thus penalizing larger errors more heavily. Root-Mean-Squared Error (RMSE) is the square root of the MSE, providing an error value in the same units as the housing prices. MAE reflects the average error magnitude, MSE highlights the impact of large deviations, and RMSE is useful for interpreting error size in original terms.	**/5**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	$\text{Gradability}_{\text{stem}}$	$\text{Gradability}_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
ZS+B *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
MZS *(rank 5):*									
1.0	1.0	0.5	1.0	1.0	0.0	1.0	0.5	1.0	0.4

Selection Justification

Representative matched-task selection: task selected from the easy difficulty pool (n=11 candidates; selected percentile = 45.5) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.8527

Blind human evaluation ranking: ZS+B > MZS+B > OURS > ZS > MZS. ZS+B was ranked first because it strikes the best balance between scenario use, conceptual value, and internal consistency: it poses a clean understanding-level task, uses a realistic regression scenario proportionately, and provides a clear, complete, and highly objective mark scheme. **MZS** was ranked last because it is Bloom-misaligned and cognitively broader than its label suggests: it combines numerical calculation with critique of metric suitability in an imbalanced disease-detection setting while being declared only as knowledge.

Figure 24: **Representative SBQ sample, easy difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Representative SBQ Sample — Medium Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 2–3 — Introduction to Machine Learning & Unsupervised Learning (Annotated Notes) | *Model:* GPT-4o-mini | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Medium

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (ZS)	Explain why a model that always predicts "no disease" can achieve 99% accuracy on this problem but is not useful. What alternative metrics should be considered to properly evaluate the model's effectiveness?	Application	Understanding†	8	0.813
B (OURS)	Differentiate between the evaluation approaches used for the classification task and the regression task described in the scenario.	Analyze	Analyze	7	**1.000**
C (MZS+B)	Compare the implications of using MAE versus MSE as evaluation metrics for this regression model, particularly in the presence of outliers in the data.	Analyze	Analyze	7	0.861
D (MZS)	Explain why the high accuracy achieved by this model might be misleading in evaluating its performance. Which alternative metrics would provide better insight into the model's effectiveness at detecting disease D, and why?	Application	Analyze†	8	0.973
E (ZS+B)	Analyse which metric, precision or sensitivity, would highlight the weakness of the algorithm that always predicts "no disease", and justify why.	Analyze	Analyze	4	0.840

†Declared–true Bloom mismatch identified by blind human evaluator.

Scenario (OURS — rank 1):

A supervised learning algorithm is developed to predict the presence of a rare disease D using chest X-ray images. The data is split into a training set, which includes labeled examples, and a test set, where labels are hidden during evaluation. The disease occurs in 1% of the population represented in the dataset. Upon evaluation, an accuracy of 99% was recorded on the test set. The test set maintains the same class distribution as the population, where the positive class is rare. Additionally, alternative classification metrics such as precision, sensitivity, and F1 score are considered for further analysis. In a separate regression task, predictions consist of real-valued outputs compared against true target values using error metrics including Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root-Mean-Squared Error (RMSE).

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Classification uses discrete class labels with metrics like accuracy, precision, sensitivity, and F1 score to evaluate prediction quality.	2
Accuracy can be misleading in classification when classes are imbalanced, such as a rare disease with 1% prevalence.	1
Regression evaluates continuous outputs using error metrics like MAE, MSE, and RMSE to measure prediction errors.	2
Classification metrics assess categorical correctness and balance of false positives/negatives, while regression metrics quantify numerical deviations.	2
Model answer: Classification uses discrete class labels with metrics like accuracy, precision, sensitivity, and F1 score to evaluate prediction quality. Accuracy can be misleading in classification when classes are imbalanced, such as a rare disease with 1% prevalence. Regression evaluates continuous outputs using error metrics like MAE, MSE, and RMSE to measure prediction errors. Classification metrics assess categorical correctness and balance of false positives and negatives, while regression metrics quantify numerical deviations.	**/7**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	Gradability$_{stem}$	Gradability$_{triple}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	1.0	1.0	1.0
ZS *(rank 5):*									
1.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.8

Selection Justification

Representative matched-task selection: task selected from the medium difficulty pool (n=30 candidates; selected percentile = 50.0) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.8974

Blind human evaluation ranking: OURS > ZS+B > MZS > MZS+B > ZS. OURS was ranked first because it is the most internally consistent, scenario-grounded, and cognitively well-targeted item in the set: it asks for a genuine analyze-level comparison, makes full use of both the classification and regression parts of the scenario, and supports highly consistent grading. **ZS** was ranked last because it overstates the cognitive level: despite a useful scenario, the question mainly asks for conceptual explanation of why 99% accuracy is misleading and which alternative metrics matter, making it closer to understanding than true application.

Figure 25: **Representative SBQ sample, medium difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies the true cognitive demand, as identified by the blind human evaluator.

Representative SBQ Sample — Hard Difficulty

Course: 4CCSAIAI Introduction to Artificial Intelligence | *Source:* Week 4 — Ethics (Slides) | *Model:* Gemini-2.5-Flash | *Task type:* Scenario-Based Question (SBQ) | *Difficulty target:* Hard

Pipeline	Question stem	Declared Bloom	True Bloom	Marks	σ_w
A (MZS+B)	Identify and explain at least two distinct causes of bias (from "World bias", "Representation bias", "Measurement bias", "Algorithm bias", "Evaluation bias") that likely contributed to the problematic auto-crop behavior observed in Rui's system.	Analyze	Analyze	6	0.965
B (ZS)	Identify and explain two significant ethical harms that the Epimetheus app could cause, and propose two distinct mitigation strategies the developers could implement to address these harms, particularly concerning personal identity.	Evaluation	Analyze†	8	0.938
C (OURS)	Develop a structured set of guidelines for future AI feature development at the social media company, explicitly addressing the ethical concerns (stereotypes, indirect discrimination, objectification) identified in the auto-crop feature's review.	Synthesis	Synthesis	12	**0.965**
D (ZS+B)	From an AI ethics perspective, explain two potential societal harms that "Sentinel-Eye" could inflict, even if it successfully reduces overt crime.	Analyze	Understanding†	4	0.849
E (MZS)	What specific measures can the developers of Epimetheus implement to address the issue of gender misidentification and its associated dignitary harms, particularly for transgender individuals, as discussed in the course material?	Evaluation	Application†	8	0.910

†Declared–true Bloom mismatch identified by blind human evaluator.

Scenario (OURS — rank 1):

A software engineer at a social media company developed an 'auto-crop' feature that automatically cropped images attached to posts. The system was configured to identify and focus on specific image areas. A deep neural network was trained on a dataset containing images and their associated saliency data. Following its deployment, the feature led to an increase in posts containing images and an uplift in user engagement metrics, including time spent on the platform and the number of clicks. However, subsequent observations by a group of users showed systematic patterns within the auto-crop's behavior. These patterns included a consistent focus on White individuals over Black individuals, famous men over famous women, and a frequent centering on women's chest areas. An internal ethics review stated that this behavior exhibited characteristics concerning stereotypes, indirect discrimination, and the objectification of women.

Mark scheme and expected answer (OURS — rank 1):

Marking point	Marks
Mandate a rigorous, pre-deployment audit process for all AI training datasets, ensuring they are demographically representative and balanced to actively prevent the emergence of stereotypes and indirect discrimination.	3
Implement ethically-guided design principles for attention or saliency mechanisms in image processing AI, specifically to avoid hyper-focus on sensitive body areas and prevent objectification, perhaps using constrained optimization or negative exemplars.	3
Integrate a comprehensive bias detection and mitigation framework into the AI development lifecycle, applying quantitative fairness metrics (e.g., demographic parity, equalized odds) at multiple stages from data preparation to model deployment.	2
Establish a standing, interdisciplinary ethical AI review board, comprising engineers, ethicists, and diverse user representatives, responsible for assessing the societal impact and potential biases of new AI features from conception.	2
Develop transparent channels for user feedback specifically on AI behavior, coupled with a swift, accountable system for investigating, communicating, and rectifying identified ethical violations in deployed features.	2
Model answer: Future AI feature development will mandate a rigorous, pre-deployment audit process for all AI training datasets, ensuring they are demographically representative and balanced to actively prevent the emergence of stereotypes and indirect discrimination. Ethically-guided design principles will be implemented for attention or saliency mechanisms in image processing AI, specifically to avoid hyper-focus on sensitive body areas and prevent objectification, perhaps using constrained optimization or negative exemplars. A comprehensive bias detection and mitigation framework will be integrated into the AI development lifecycle, applying quantitative fairness metrics like demographic parity or equalized odds at multiple stages from data preparation to model deployment. Furthermore, a standing, interdisciplinary ethical AI review board, comprising engineers, ethicists, and diverse user representatives, will be established, responsible for assessing the societal impact and potential biases of new AI features from conception. Finally, transparent channels for user feedback specifically on AI behavior will be developed, coupled with a swift, accountable system for investigating, communicating, and rectifying identified ethical violations in deployed features.	**/12**

Per-criterion scores (best and worst pipeline):

Grounding	Clarity	Integrity	$\text{Gradability}_{\text{stem}}$	$\text{Gradability}_{\text{triple}}$	Mk. scheme	Fidelity	SRN	SG	BLA
OURS *(rank 1):*									
1.0	1.0	1.0	1.0	1.0	**1.0**	1.0	0.5	1.0	1.0
ZS+B *(rank 5):*									
1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.6

Selection Justification

Representative matched-task selection: task selected from the hard difficulty pool (n=16 candidates; selected percentile = 50.0) as a representative matched-task sample rather than an extreme contrast case. **Mean σ_w:** 0.9252

Blind human evaluation ranking: OURS > MZS+B > ZS > MZS > ZS+B. OURS was ranked first because it best combines scenario use, cognitive demand, and assessment structure: the auto-crop case is essential rather than decorative, the task requires genuine synthesis, and the mark scheme is both concrete and flexible enough to support higher-order ethical reasoning. **ZS+B** was ranked last because, although clear and grounded, it is more explanatory than analytical: asking for two harms of Sentinel-Eye is better characterized as understanding than true analysis, making it less discriminative as a representative hard SBQ.

Figure 26: **Representative SBQ sample, hard difficulty tier.** Pipeline identities are shown explicitly: ZS = zero_shot, ZS+B = zero_shot_bloom, MZS = multi_stage_zero_shot, MZS+B = multi_stage_zero_shot_bloom, and Ours = proposed pipeline. σ_w denotes the discrimination-weighted composite criterion score (higher is better, max = 1). The mark scheme and per-criterion breakdown are shown for the rank-1 and rank-5 pipelines only. †Flags items where the declared Bloom level under-specifies or overstates the true cognitive demand, as identified by the blind human evaluator.

Evaluation Type	QType	N	Prop.	Base.	Rate
Representative	SAQ	3	2	1	66.7%
Representative	SBQ	3	2	1	66.7%
Illus. Contrast (msg)	SAQ	3	2	1	66.7%
Illus. Contrast (msg)	SBQ	3	2	1	66.7%
Illus. Contrast (bbg)	SAQ	3	3	0	100%
Illus. Contrast (bbg)	SBQ	3	2	1	66.7%
Overall	All	**18**	**13**	**5**	**72.2%**

Table 14: Aggregate qualitative selection outcomes across all 18 matched-task manual comparisons. msg = max_score_gap; bbg = bloom_bla_gap. "Prop." and "Base." count the number of settings in which the proposed pipeline or any baseline ranked first respectively.

Difficulty	1st	2nd	3rd	4th	5th
Easy	**Ours**	ZS+B	MZS+B	ZS	MZS
Medium	**Ours**	MZS+B	ZS+B	MZS	ZS
Hard	MZS+B	**Ours**	ZS+B	ZS	MZS

Table 15: Pipeline rankings for representative SAQ evaluations by difficulty level.

Difficulty	1st	2nd	3rd	4th	5th
Easy	ZS+B	MZS+B	**Ours**	ZS	MZS
Medium	**Ours**	ZS+B	MZS	MZS+B	ZS
Hard	**Ours**	MZS+B	ZS	MZS	ZS+B

Table 16: Pipeline rankings for representative SBQ evaluations by difficulty level.

Difficulty	1st	2nd	3rd	4th	5th
Easy	**Ours**	ZS+B	MZS+B	ZS	MZS
Medium	ZS+B	**Ours**	MZS+B	MZS	ZS
Hard	**Ours**	MZS+B	ZS	MZS	ZS+B

Table 17: Pipeline rankings for SAQ illustrative contrasts on the max_score_gap axis.

Difficulty	1st	2nd	3rd	4th	5th
Easy	**Ours**	ZS	MZS+B	ZS+B	MZS
Medium	**Ours**	ZS+B	MZS	ZS	MZS+B
Hard	**Ours**	MZS+B	ZS	MZS	ZS+B

Table 18: Pipeline rankings for SAQ illustrative contrasts on the bloom_bla_gap axis.

Difficulty	1st	2nd	3rd	4th	5th
Easy	MZS+B	ZS+B	**Ours**	ZS	MZS
Medium	**Ours**	ZS	ZS+B	MZS	MZS+B
Hard	**Ours**	MZS+B	ZS	ZS+B	MZS

Table 19: Pipeline rankings for SBQ illustrative contrasts on the max_score_gap axis.

Difficulty	1st	2nd	3rd	4th	5th
Easy	MZS+B	ZS+B	**Ours**	ZS	MZS
Medium	**Ours**	ZS+B	ZS	MZS	MZS+B
Hard	**Ours**	ZS+B	MZS+B	ZS	MZS

Table 20: Pipeline rankings for SBQ illustrative contrasts on the bloom_bla_gap axis.