Cognitive Episodes in LLM Reasoning Traces Enable Interpretable Human Item Difficulty Prediction

Chenguang Wang
Virginia Tech
cswang@vt.edu

Ming Li
University of Maryland and MBZUAI
minglii@umd.edu

Xinyue Zeng
Virginia Tech

Zhuochun Li
University of Pittsburgh

Hong Jiao
University of Maryland

Tianyi Zhou
MBZUAI
tianyi.zhou@mbzuai.ac.ae

Dawei Zhou
Virginia Tech
dzhou@vt.edu

Abstract

Predicting human item difficulty is a central challenge in educational assessment, where accurate estimates are critical for fairness and effective test construction. Existing approaches typically rely on costly human calibration or item-level textual representations, offering limited interpretable evidence about the cognitive processes that make an item difficult. We argue that item difficulty should be viewed not only as a property of item text, but also as an **observable consequence of the problem-solving burden induced by the item**. Yet, this burden from human is difficult to observe directly or analyze at scale. Reasoning-oriented large language models (LLMs) offer scalable process evidence through explicit reasoning traces, but raw traces are often lengthy and unstructured. To organize this evidence, we introduce cognitive episodes, grounded in Schoenfeld's account of problem solving, which group trace segments into functional problem-solving states. Building on this representation, we propose **Epi2Diff** (Episode to Difficulty), a framework that converts model-generated reasoning traces into episode sequences and extracts compact features capturing the scale, allocation, and transitions of reasoning effort. These process features are combined with semantic item representations for human item difficulty prediction. Experiments on four real-world datasets show that Epi2Diff consistently outperforms strong baselines, including fine-tuned SLMs, LLM in-context learning, and supervised LLM adaptation. On the SAT-derived classification benchmarks, Epi2Diff achieves an 8.1% average relative gain over supervised LLM fine-tuning baselines. Further analyses show that harder items induce more effortful, iterative, and implementation-centered episode dynamics rather than merely longer responses. These results establish cognitive episodes as a predictive and interpretable process representation for human item difficulty. The anonymized code is available at https://anonymous.4open.science/r/Epi2Diff-B10A.

1 Introduction

Large-scale assessment is a core evaluation mechanism in domains such as college admissions, language testing, and medical licensure (Hambleton et al., 1991; Eignor, 2013; Hsu et al., 2018; AlKhuzaey et al., 2021). In these settings, item difficulty prediction is important for constructing assessments with well-calibrated difficulty distributions and precise measurement along the ability scale (AlKhuzaey et al., 2021; Parry, 2020). Traditionally,

item difficulty has been estimated from examinee responses using classical psychometric frameworks such as Classical Test Theory (CTT) and Item Response Theory (IRT) from responses collected during field testing (Hsu et al., 2018; DeMars, 2010; AlKhuzaey et al., 2024; Benedetto, 2023). Although effective, these approaches require substantial pretesting and are therefore costly, time-consuming, and difficult to scale.

In response to these limitations, prior work has explored automated approaches for predicting item difficulty. Existing approaches mainly include feature-based methods and end-to-end fine-tuning methods (Peters et al., 2025; AlKhuzaey et al., 2024; Benedetto, 2023). Feature-based methods are generally more interpretable but depend on representation quality, whereas end-to-end fine-tuning methods, such as fine-tuning Small Language Models (SLMs) like BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019), are more streamlined but often less interpretable and more vulnerable in small-data settings (Peters et al., 2025; Rogoz & Ionescu, 2024; Benedetto, 2023; Tack et al., 2024; Xue et al., 2020). This is particularly problematic in educational assessment, where both accuracy and interpretability are needed (Messick, 1990; Kane, 2013). More recently, LLMs, including reasoning-oriented models, have enabled new strategies for item difficulty prediction, including inference-time reasoning, fine-tuning, LLM-extracted features, reasoning- or sampling-augmented prediction, and uncertainty-based signals (Rogoz & Ionescu, 2024; Dueñas et al., 2024; Razavi & Powers, 2025; Feng et al., 2025a; Zotos et al., 2024; Li et al., 2025c). While these methods show that LLMs can provide useful signals for item difficulty prediction, their effectiveness depends on how such signals are elicited and modeled. Further, they often rely on model-generated outputs as predictive cues without explicitly grounding them in interpretable evidence aligned with human psychometric item difficulty. Meanwhile, a related but distinct line of work has operationalized difficulty in broader LLM settings, such as by stratifying data for easy-to-hard or cross-difficulty transfer or by scaling benchmark difficulty through required reasoning length (Hase et al., 2024; Kordi et al., 2026; Ding et al., 2025). However, these studies use difficulty mainly to analyze LLM behavior, rather than to transform reasoning processes into interpretable evidence for predicting human item difficulty.

Collectively, these limitations suggest that **the key challenge for existing approaches lies not only in model capacity, but also in how human item difficulty is treated and modeled in prediction.** Across much of the text-based item-difficulty literature, difficulty is treated as a single item-level outcome and predicted from item text, metadata, annotations, embeddings, or other model-derived representations (Benedetto, 2023; AlKhuzaey et al., 2024). Although this formulation is psychometrically natural, it leaves open how to represent the cognitive and process-level factors that may be associated with observed difficulty. Educational psychology suggests that such factors may include cognitive demand or problem-solving burden, such as coordination of operations, representation management, monitoring, verification, and effortful solution processes (Sweller, 1988; 2011; Noroozi & Karami, 2022; Schoenfeld, 2016).

The challenge is that such cognitive demands underlying the item difficulty have historically been difficult to observe and model at scale. In assessment research, they are typically studied through indirect and labor-intensive methods such as think-aloud protocols and eye-tracking. Recent reasoning models provide a new opportunity to operationalize this perspective because they often produce explicit reasoning traces for complex problem solving (Guo et al., 2025; Jaech et al., 2024; Xiong et al., 2025; Muennighoff et al., 2025). In particular, they can expose process categories such as decomposition, implementation, revision, and verification that are largely hidden in response-only outputs from instruction-following LLMs and can be organized using cognitively grounded accounts of problem solving. Although these traces should not be treated as direct observations of human cognition, they can serve as scalable, model-generated proxies for solution processes.

However, translating this insight into an effective model is nontrivial, since model-generated reasoning traces are often long, redundant, and noisy. In item difficulty prediction, richer or higher-dimensional representations do not necessarily yield better predictions (Fulari & Rusert, 2024; Tack et al., 2024). Directly incorporating raw reasoning traces may therefore introduce irrelevant information and even degrade performance. Accordingly, the central

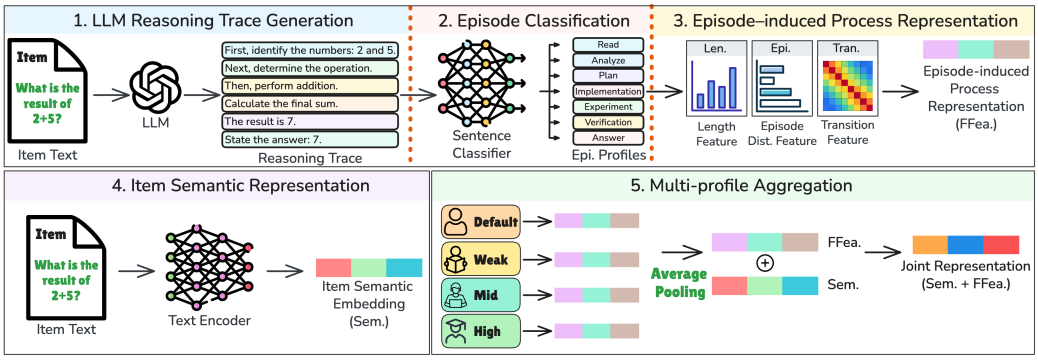


Figure 1: **Overview of Epi2Diff.** The framework generates reasoning traces with reasoning-oriented LLMs, maps them to sentence-level episode profiles, builds episode-induced process representations from length, episode distribution, and transition features, and combines them with item semantic embeddings through multi-profile aggregation for item difficulty prediction.

question of this work is: ***How can explicit model reasoning processes be transformed into a structured and informative representation for modeling item difficulty?***

To tackle this challenge, we advance item difficulty prediction by augmenting semantic item representations with cognitive episode signals derived from model-generated reasoning traces. Specifically, **Epi2Diff** (Episode to Difficulty) converts model-generated reasoning traces into sentence-level cognitive episode sequences using a cognitively grounded episode taxonomy (Schoenfeld, 2014; 2016; Li et al., 2025d;b). These sequences provide a structured and interpretable view of reasoning dynamics. From these sequences, we derive compact process features that capture the scale, composition, and organization of the reasoning process, and combine them with semantic embeddings of item content to form the final representation for downstream difficulty prediction. This representation is valuable not only for improving predictive accuracy but also for supporting interpretable analyses of item difficulty by identifying episode types and transition patterns associated with greater difficulty, such as prolonged implementation, repeated verification, or inefficient local transitions. Experiments on four datasets spanning SAT, Cambridge English Qualifications, and USMLE show that our method consistently outperforms various baselines, including fine-tuning small encoder-only language models, LLM In-Context Learning, and LLM Supervised Fine-tuning with full parameters or LoRA (Hu et al., 2022). On the real-world SAT-derived classification benchmarks, Epi2Diff achieves an 8.1% average relative gain over LLM supervised fine-tuning baselines, while also improving regression performance on Cambridge English Qualifications and USMLE.

Therefore, our contributions are threefold:

- We extend human item difficulty prediction beyond text representations by **modeling difficulty as an observable consequence of item-induced problem-solving burden**, using model-generated reasoning traces as scalable proxy evidence.
- We develop a cognitively grounded episode representation that captures the scale, allocation, and flow of reasoning effort, enabling both accurate prediction and process-based interpretation.
- Across four real-world assessment benchmarks, Epi2Diff consistently outperforms strong baselines in classification and regression; further analyses show that harder items elicit more effortful, implementation-centered, and iterative reasoning dynamics rather than merely longer traces.

2 Method

2.1 Preliminary

2.1.1 *Schoenfeld's Episode Theory*

Schoenfeld's Episode Theory (Schoenfeld, 2014) models the problem-solving process as a temporally ordered sequence of functional episodes. Schoenfeld's framework comprises seven episodes: six original episodes, *Read*, *Analyze*, *Plan*, *Implement*, *Explore*, and *Verify*, plus the later-added *Monitor* episode. This framework is well-suited to our setting because it provides a cognitively grounded vocabulary for characterizing how problem-solving burden is distributed across different stages of long reasoning traces. Following ThinkARM (Li et al., 2025b), we add an *Answer* episode to capture final answer commitment, which is often distinct from preceding verification and monitoring. We therefore use an eight-way taxonomy consisting of *Read*, *Analyze*, *Plan*, *Implement*, *Explore*, *Verify*, *Monitor*, and *Answer*.

2.1.2 *Problem Formulation*

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a dataset of assessment items, where $x_i \in \mathcal{X}$ is a raw assessment item and $y_i \in \mathcal{Y}$ is its ground-truth difficulty label. Depending on the label form, we consider either classification with $\mathcal{Y} = \{\text{Easy}, \text{Medium}, \text{Hard}\}$ or regression with $\mathcal{Y} \subseteq \mathbb{R}$.

For each item x_i, we query a reasoning-oriented LLM under a set of prompting conditions $\mathcal{P} = \{p_m\}_{m=1}^{M}$. Under condition p_m, the rollout operator Φ produces a reasoning rollout $r_i^{(m)} = \Phi(x_i, p_m) = \big(\tau_i^{(m)}, \hat{a}_i^{(m)}\big)$, and the resulting rollout set is $\mathcal{R}_i = \{r_i^{(m)}\}_{m=1}^{M}$. We then construct an item representation from x_i and $\mathcal{R}_i$ and predict difficulty via

$$\hat{y}_i = f\big(S(x_i, \mathcal{R}_i)\big), \qquad y_i \in \mathcal{Y}, \tag{1}$$

where S maps the item and its associated reasoning rollouts into a representation combining item-side semantic information and process signals derived from the reasoning traces, and f is a classifier or regressor depending on the label space.

The model is trained by minimizing

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell(\hat{y}_i, y_i), \tag{2}$$

where ℓ is the task-specific loss, e.g., cross-entropy for classification and mean squared error for regression.

2.2 Sentence-Level Episode Profiling from Reasoning Traces

To support sentence-level episode profiling, we build a labeled corpus from the ThinkARM reasoning-trace annotations. ThinkARM is constructed from reasoning traces collected on Omni-MATH (Gao et al., 2024) and annotated at the sentence level under the episode taxonomy introduced earlier. In this work, we use the subset generated by five representative reasoning models, including DeepSeek-R1 (Guo et al., 2025), R1-Distill-Qwen3-32B (Guo et al., 2025), QwQ-32B (Team, 2025), Qwen3-32B (Yang et al., 2025), and Phi4-Reasoning (Abdin et al., 2024b). Each reasoning trace is segmented into sentences using a standard sentence-splitting pipeline, and each sentence is paired with its corresponding episode label. The annotation resources include both large-scale GPT-5 annotations and a human-verified gold set of 7,067 annotated sentences. We use the GPT-5-annotated data for training and validation, and reserve the human-annotated gold set for final testing. Since our goal is sentence-level episode prediction, we flatten the reasoning traces into a unified pool of sentence, episode pairs. After basic cleaning to remove invalid samples, reduce label inconsistencies, and deduplicate repeated pairs, the final corpus contains 231,913 labeled sentence, episode pairs for training and validation.

We train a RoBERTa-based sentence classifier on the cleaned corpus for eight-way episode prediction. On the human-annotated gold test set, the classifier achieves an accuracy of

0.799, a macro-F1 of 0.761, and a micro-F1 of 0.799. We then use the trained classifier as an automatic episode labeler in our following method. For each reasoning trace, we apply the same sentence segmentation pipeline and predict an episode label for every sentence, thereby inducing an episode-level representation for each item.

2.3 Item Representation

Prior work has shown that reasoning traces exhibit structured episode-level dynamics, suggesting that the problem-solving burden of an item may be reflected in both episode-wise properties and inter-episode transitions. Building on this insight, we construct an episode-based representation from labeled reasoning traces to model such burden and improve item difficulty prediction. To determine the most effective representation construction, we systematically explored episode-statistic features at multiple levels, ranging from basic episode distributions to multi-hop transition patterns, and empirically evaluated their individual contributions and combinations in downstream item difficulty prediction.

Based on this feature search and performance comparison, we adopt a two-part item representation: (1) an item semantic representation, which encodes the item content; and (2) episode-induced process representation for capturing the problem-solving burden reflected in reasoning traces. Concretely, the episode-induced process representation combines three complementary groups of features: length features capturing the overall scale of the reasoning process, episode-distribution features characterizing its composition across episode types, and transition features encoding its local sequential organization through inter-episode transitions.

2.3.1 Episode-induced process representation

For each item, we represent its reasoning trace as a labeled episode sequence over ordered sentence-level units and derive the episode-induced process representation from it.

Length features We first construct three length features for each item from its labeled reasoning trace, corresponding to the total token length of the trace, the token length of the thinking part, and the token length of the answer part. Here, the `think/answer` distinction follows the structure of the model response, where the explicit reasoning-trace portion is separated from the final-answer portion, and all lengths are measured in tokens. We denote these three quantities by $l_{i,\text{total}}$, $l_{i,\text{think}}$, and $l_{i,\text{answer}}$, and group them into a 3-dimensional length vector $\mathbf{L}_i = [l_{i,\text{total}}; l_{i,\text{think}}; l_{i,\text{answer}}] \in \mathbb{R}^3$. The resulting vector captures the overall scale of the reasoning process while distinguishing reasoning effort from final answer production.

Episode-distribution features We next characterize how the reasoning budget is distributed across episode types. For each episode $e \in \mathcal{E}$, we compute an episode token count $c_{i,e}$ as the total token length of all labeled sentence-level units assigned to episode e. We then normalize these episode counts to obtain episode ratios:

$$r_{i,e} = \frac{c_{i,e}}{\sum_{e' \in \mathcal{E}} c_{i,e'}}. \tag{3}$$

Based on these quantities, we form an episode-count vector $\mathbf{C}_i = [c_{i,e}]_{e \in \mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}$ and an episode-ratio vector $\mathbf{R}_i = [r_{i,e}]_{e \in \mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}$. The count vector preserves the absolute amount of trace content allocated to each episode, whereas the ratio vector captures the relative allocation of reasoning effort after normalizing for total episode token mass. The two vectors jointly preserve the overall reasoning scale and enable comparison across traces.

Transition features Finally, we characterize reasoning flow through episode transitions. Rather than treating a trace as an unordered collection of episode statistics, we record how often it moves from one episode type to the next along the labeled sequence. For any pair of episodes $(e, e') \in \mathcal{E} \times \mathcal{E}$, we define a transition count $a_{i,e,e'}$ as the number of times episode e is immediately followed by episode e'. Collecting all such counts yields an episode

transition matrix $\mathbf{A}_i = [a_{i,e,e'}]_{e,e' \in \mathcal{E}} \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}|}$. In our setting, $|\mathcal{E}| = 8$, so $\mathbf{A}_i$ is an 8×8 matrix whose entries record raw adjacent transition counts.

This matrix captures local sequential organization that is not available from token totals alone. For example, two traces may allocate similar reasoning effort across episode types but still differ substantially in how they move through them over time. To combine transition features with the other process feature groups, we flatten $\mathbf{A}_i$ into a 64-dimensional vector and concatenate it with the length, episode-count, and episode-ratio features. The resulting episode-induced process representation is therefore $\mathbf{p}_i = [\mathbf{L}_i; \mathbf{C}_i; \mathbf{R}_i; \mathrm{vec}(\mathbf{A}_i)] \in \mathbb{R}^{83}$, where the 83 dimensions consist of 3 length features, 8 episode token counts, 8 episode token ratios, and 64 transition counts.

2.3.2 Item Semantic Representation and Joint Representation

Item semantic representation. Episode-induced process representations characterize the solution processes elicited from the selected reasoning models, but they do not directly encode the semantic content of the item itself. We therefore complement $\mathbf{p}_i$ with an item semantic representation obtained from a Sentence-BERT encoder (Reimers & Gurevych, 2019). Specifically, we encode the item text into a contextual embedding $\mathbf{h}_i \in \mathbb{R}^d$, which captures its semantic meaning in a dense continuous space. This embedding provides information about the content and formulation of the item that is not directly reflected in reasoning-trace statistics.

Joint representation. We then combine the semantic and process components to form the joint representation, denoted by $\mathbf{u}_i = [\mathbf{h}_i; \mathbf{p}_i]$. The item semantic representation $\mathbf{h}_i$ captures the content of the item, whereas the Episode-induced process representation $\mathbf{p}_i$ captures the reasoning burden and structural dynamics elicited by it. The resulting joint representation, therefore, integrates both what the item is about and how it is solved.

2.4 Item Difficulty Prediction

Given an item, we further construct its final representation by aggregating reasoning processes induced under multiple proficiency-conditioned model profiles (Yuan et al., 2026). Inspired by psychological assessment frameworks that consider different ability levels, we use direct, weak, medium, and strong prompting conditions to elicit diverse solution processes from the reasoning model. These profiles are prompting conditions and should not be interpreted as validated simulations of human students or ability levels. Let $s \in \mathcal{S}$ index the model profile, where $\mathcal{S} = \{\text{direct}, \text{weak}, \text{medium}, \text{strong}\}$. For each profile s, we obtain a reasoning trace $\tau_i^{(s)}$ and map it to an episode-induced process representation $\mathbf{p}_i^{(s)}$. We then average these process representations across model profiles:

$$\bar{\mathbf{p}}_i = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathbf{p}_i^{(s)}. \tag{4}$$

Since the underlying item content is identical across model profiles, the semantic representation $\mathbf{h}_i$ is kept fixed and is not updated during this stage. The final item representation is then formed as $\mathbf{u}_i = [\mathbf{h}_i; \bar{\mathbf{p}}_i]$.

We then train supervised predictors on top of the final item representation for difficulty estimation. All predictors are trained and evaluated under the same train/validation/test split protocol, and preprocessing statistics are computed using only the training split to avoid data leakage. For categorical difficulty estimation, we instantiate the predictor as a classifier; for continuous difficulty estimation, we instantiate it as a regressor. To evaluate the robustness of the proposed representation across different inductive biases, we experiment with both linear and tree-based predictors. For classification, we use Logistic Regression, Random Forest, and XGBoost. For regression, we use Ridge, Random Forest, and XGBoost. For scale-sensitive linear models, the input representation is standardized using training-split statistics, i.e., $\tilde{\mathbf{u}}_i = (\mathbf{u}_i - \boldsymbol{\mu})/\boldsymbol{\sigma}$, while tree-based models are trained on the original feature scale. This formulation allows us to evaluate whether the proposed

Method	SAT Math		SAT Reading		Cambridge		USMLE	
	ACC ↑	F1 ↑	ACC ↑	F1 ↑	RMSE ↓	R^2 ↑	RMSE ↓	R^2 ↑
SLM Fine-tuning								
BERT (Devlin et al., 2019)	0.619	0.622	0.619	0.617	8.215	0.296	0.302	0.052
RoBERTa (Liu et al., 2019)	0.601	0.602	0.623	0.618	8.311	0.280	0.298	0.076
ModernBERT (Warner et al., 2024)	0.637	0.643	0.601	0.598	8.200	0.299	0.306	0.026
ELECTRA (Clark et al., 2020)	0.609	0.604	0.623	0.622	8.277	0.285	0.304	0.043
LLM Zero-shot Inference and In-Context Learning								
GPT-4o (0-shot) (OpenAI et al., 2024)	0.447	0.350	0.478	0.397	13.693	-0.956	0.347	-0.246
GPT-5 (0-shot) (Singh et al., 2025)	0.349	0.188	0.381	0.225	33.657	-10.815	0.365	-0.380
QwQ-32B (0-shot) (Team, 2025)	0.405	0.282	0.477	0.384	32.022	-9.694	0.349	-0.165
Qwen3-32B (0-shot) (Yang et al., 2025)	0.449	0.304	0.440	0.349	19.376	-2.916	0.389	-0.509
GPT-4o (3-shot) (OpenAI et al., 2024)	0.391	0.376	0.246	0.255	24.518	-5.217	0.358	-0.357
GPT-5 (3-shot) (Singh et al., 2025)	0.358	0.212	0.384	0.230	23.026	-4.529	0.350	-0.269
QwQ-32B (3-shot) (Team, 2025)	0.321	0.321	0.011	0.020	26.787	-6.102	0.399	-0.569
Qwen3-32B (3-shot) (Yang et al., 2025)	0.405	0.359	0.299	0.335	16.371	-1.853	0.393	-0.559
GPT-4o (5-shot) (OpenAI et al., 2024)	0.381	0.378	0.291	0.301	16.181	-1.758	0.353	-0.272
GPT-5 (5-shot) (Singh et al., 2025)	0.349	0.202	0.381	0.225	19.540	-2.964	0.351	-0.277
QwQ-32B (5-shot) (Team, 2025)	0.237	0.264	0.022	0.042	25.537	-5.752	0.402	-0.647
Qwen3-32B (5-shot) (Yang et al., 2025)	0.391	0.374	0.235	0.285	15.122	-1.500	0.513	-2.084
LLM Supervised Fine-tuning (Full Parameter and LoRA)								
Qwen2.5-3B (full) (Yang et al., 2024)	0.660	0.667	0.578	0.573	10.227	-0.107	0.478	-1.368
Qwen3-4B (full) (Yang et al., 2025)	0.698	0.705	0.619	0.618	10.242	-0.386	0.478	-1.368
Phi-3.5-mini (full) (Abdin et al., 2024a)	0.674	0.681	0.556	0.540	14.045	-1.152	0.485	-1.441
Phi-4-mini (full) (Abdin et al., 2024b)	0.684	0.691	0.619	0.615	11.055	-0.258	0.459	-1.188
Llama3.2-3B (full) (Grattafiori et al., 2024)	0.679	0.687	0.601	0.597	9.380	0.086	0.460	-1.196
Qwen2.5-3B (LoRA) (Yang et al., 2024)	0.628	0.611	0.608	0.602	13.419	-1.004	0.371	-0.431
Qwen3-4B (LoRA) (Yang et al., 2025)	0.628	0.629	0.590	0.585	10.524	-0.228	0.414	-0.775
Phi-3.5-mini (LoRA) (Abdin et al., 2024a)	0.637	0.631	0.586	0.587	12.779	-0.985	0.449	-1.093
Phi-4-mini (LoRA) (Abdin et al., 2024b)	0.647	0.648	0.541	0.536	10.523	-0.221	0.469	-1.284
Llama3.2-3B (LoRA) (Grattafiori et al., 2024)	0.595	0.591	0.590	0.590	9.117	0.076	0.450	-1.097
Epi2Diff (ours)	**0.730**	**0.728**	**0.631**	**0.626**	**7.612**	**0.396**	**0.291**	**0.121**

Table 1: **Main results** on four item difficulty prediction benchmarks: SAT Math, SAT Reading, Cambridge, and USMLE. We compare Epi2Diff with representative baselines from SLM fine-tuning, zero-shot and in-context LLM prompting, and supervised LLM adaptation under both full-parameter and LoRA settings. We report Accuracy and macro-F1 on SAT Math and SAT Reading, and RMSE and R^2 on Cambridge and USMLE. Best results are shown in bold.

episode-informed representation supports both linear and non-linear predictors under a unified setup. For each dataset, we select the downstream predictor from the candidate set above based on validation performance. Notably, across all datasets, XGBoost is consistently chosen under this protocol, so the reported results for our method are all obtained with XGBoost in the following section.

3 Experiments and Results

Details of the datasets, baselines, and evaluation metrics are provided in Appendix A.

3.1 Main Results

Table 1 reports results on four test sets spanning two classification datasets (SAT Math and SAT Reading & Writing) and two regression datasets (Cambridge and USMLE). Overall, our method achieves the best performance across all four test sets and all reported metrics, suggesting that structured representations derived from model-generated reasoning

	QwQ		Qwen		Both		QwQ (Role)		Qwen (Role)		Both (Role)	
Method	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**
Sem. + FFea.	**0.684**	**0.681**	**0.712**	**0.708**	**0.707**	**0.708**	**0.698**	**0.697**	**0.712**	**0.712**	**0.730**	**0.728**
Sem. Only	0.609	0.607	0.609	0.607	0.609	0.607	0.609	0.607	0.609	0.607	0.609	0.607
FFea. Only	0.670	0.670	0.674	0.674	0.684	0.677	0.614	0.618	0.660	0.657	0.637	0.631
Len. Only	0.540	0.539	0.502	0.493	0.553	0.555	0.530	0.529	0.558	0.553	0.553	0.551
Epi. Only	0.581	0.580	0.614	0.609	0.633	0.624	0.591	0.592	0.637	0.636	0.628	0.621
Tran. Only	0.628	0.624	0.595	0.575	0.647	0.639	0.577	0.568	0.619	0.616	0.614	0.604
Sem. w/o Len.	0.642	0.642	0.643	0.647	0.674	0.670	0.660	0.663	0.665	0.669	0.702	0.703
Sem. w/o Epi.	0.651	0.652	0.651	0.647	0.656	0.646	0.665	0.666	0.679	0.679	0.688	0.690
Sem. w/o Tran.	0.647	0.647	0.679	0.676	0.684	0.684	0.670	0.673	0.688	0.689	0.674	0.676
Sem. + Len.	0.637	0.635	0.656	0.652	0.670	0.664	0.660	0.660	0.679	0.678	0.670	0.671
Sem. + Epi.	0.656	0.652	0.674	0.670	0.674	0.672	0.660	0.661	0.684	0.682	0.684	0.686
Sem. + Tran.	0.642	0.641	0.665	0.663	0.660	0.656	0.647	0.644	0.688	0.688	0.698	0.693

Table 2: **Ablation results on SAT Math** across reasoning-trace sources and feature combinations. *QwQ* and *Qwen* denote the reasoning model that generated the traces, while *Both* denotes averaging features from both trace sources. *Role* indicates averaging features across proficiency-conditioned model profiles. *Sem.* denotes the semantic embedding, *FFea.* denotes the full process-feature set comprising length, episode-distribution, and transition features, *Len.* denotes the length feature, *Epi.* denotes the episode-distribution feature, and *Tran.* denotes the transition feature. Additional results are in Table 7.

traces provide more informative signals for item difficulty prediction than conventional approaches.

For classification, our method performs best on both SAT datasets, with especially clear gains on SAT Math. This suggests that reasoning-process features are particularly useful when difficulty depends on multi-step problem solving, while the improvement on SAT Reading & Writing indicates that the proposed representation also generalizes to text-based assessment items. A similar trend appears in regression. Our method achieves the strongest results on both Cambridge and USMLE, outperforming the best baseline models in terms of both error reduction and goodness of fit. In contrast, several prompting-based and LLM fine-tuning baselines perform poorly on these regression tasks, with some even producing negative R^2 values. This suggests that our method is especially beneficial for continuous difficulty estimation.

Overall, the results show that simply scaling model capacity does not reliably improve item difficulty prediction. By contrast, combining reasoning-trace-based process features with semantic item representations yields consistent gains across task types and domains, supporting our hypothesis that item difficulty is better captured by the structure of the underlying problem-solving process than by item text or direct model judgments alone.

3.2 Ablation Study

To comprehensively evaluate our approach, we conduct an ablation study on the SAT Math dataset while keeping all other experimental settings fixed. The study examines two aspects: the aggregation strategy and the representation design. Results are reported in Table 2.

Aggregation strategy We evaluate increasingly rich aggregation settings, including single-source reasoning traces under direct prompting (*QwQ* and *Qwen*), source-level aggregation over the two trace generators (*Both*), and profile-based aggregation, where traces from different proficiency-conditioned model profiles are aggregated either within a single source (*QwQ (Role)* and *Qwen (Role)*) or across both sources together (*Both (Role)*). The effect of aggregation is not strictly monotonic, but richer aggregation tends to improve performance. For example, under the full representation (*Sem. + FFea.*), source-level aggregation does not improve over the stronger single-source result: the macro-F1 of *Both* is 0.708, matching *Qwen* and exceeding *QwQ* at 0.681. In contrast, profile-based aggregation is more beneficial when applied across both sources: *Both (Role)* achieves the best overall performance, reaching 0.730 accuracy and 0.728 macro-F1, compared with 0.707/0.708 for *Both*. These results

suggest that jointly aggregating diverse model profiles and trace sources provides a richer characterization of the elicited reasoning processes.

A potential concern is that profile-based aggregation benefits simply from using more rollouts. We therefore compare profiled aggregation with a matched number of repeated direct-prompt generations in Appendix C.4. Profiled aggregation remains stronger under the same rollout budget, particularly in the combined-source setting, where it achieves a macro-F1 of 0.728 compared with 0.714 for direct replication. This result suggests that the gain reflects structured profile diversity rather than additional stochastic samples alone.

Representation design. Across all aggregation settings, the full representation (*Sem. + FFea.*) consistently achieves the best performance, showing the benefit of combining semantic and process-based features. Using either semantic embeddings alone or process features alone leads to clear drops, with *Sem. Only* trailing the full model by 7.4–12.1 macro-F1 points. Removing any one process feature family also consistently hurts performance, indicating that length, episode-distribution, and transition features provide complementary signals. Their contributions vary by setting: length and episode features tend to matter more in simpler settings, while transition features are especially important in the richest aggregated setting, *Both (Role)*. Although adding any single process feature family already improves over *Sem. Only*, the full combination remains the most effective. Notably, process features alone are competitive with semantic features in several settings, but the strongest results consistently require their combination.

4 Interpreting Item Difficulty through Episode-induced Features

To better understand what episode-induced representations reveal about item difficulty beyond predictive accuracy, we use SAT Math as a testbed. We ask whether these features add predictive signal beyond item semantics and trace-length features, and what directional difficulty patterns the fitted model associates with them.

Key Finding 1: Difficulty Is Associated with a Structured Process Signature.

Harder items are associated with more intermediate work, more implementation-centered effort, and stronger analysis, implementation transition signals in model-generated reasoning traces.

Appendix C.2, Table 6 provides the main model-grounded evidence for this process signature. The largest positive standardized marginal signals come from total trace length and the *think* portion, while final *answer* length has a much smaller positive signal. This suggests that difficulty is reflected less in more verbose final responses than in additional intermediate problem-solving work. At the episode level, *Implement* has the strongest positive count signal, and its ratio signal is also positive. In contrast, the ratios of *Read*, *Plan*, and *Verify* are negative, suggesting that harder items are associated with a relative shift toward concrete solution execution rather than a proportional increase in all types of reasoning activity.

The transition signals suggest how this execution-centered burden is reflected in local trace dynamics. The strongest positive transitions include *Analyze→Implement*, *Implement→Implement*, and *Implement→Analyze*. These patterns suggest that harder items are not only associated with more implementation overall, but also with more extended and iterative local execution: higher counts of these transitions are associated with higher fitted difficulty scores, indicating movement from analysis into implementation, sustained implementation across adjacent steps, and returns from implementation to analysis. This is consistent with an analysis-implementation-refinement pattern, but we do not treat these transitions as direct evidence of a distinct verification phase. Rather, the strongest transition-level predictive signals appear to come mainly from analysis–implementation dynamics, which helps explain why the standalone *Verify* episode need not increase.

Together, these results give a compact interpretation of difficulty: length captures the *scale* of the burden, episode composition captures its *allocation*, and transitions capture its *flow*.

The main insight is therefore not that difficult items simply make the model think longer, but that they are associated with a more structured and implementation-centered model reasoning process.

Key Finding 2: Episode Dynamics Provide Predictive Process Evidence.

Episode composition and transition dynamics contribute complementary process evidence for difficulty prediction.

We next connect the structured-burden interpretation to the features used by the predictor. We assess feature-group importance in the profile-aggregated *Both (Role)* setting; Appendix C.2, Table 5 reports the complete results across trace-source and profile-aggregation settings. We interpret these feature groups as complementary views of item difficulty: *Sem.* captures item content, *Len.* captures reasoning scale, *Epi.* captures effort allocation, and *Tran.* captures local reasoning flow.

The feature-importance analysis shows that semantic features remain an important anchor, especially under SHAP importance, as expected for item difficulty prediction. At the same time, the combined episode-structural group, *Epi.+Tran.*, is also informative. In the *Both (Role)* setting, *Epi.+Tran.* reaches 0.136 in permutation importance, exceeding *Sem.* at 0.084 and *Len.* at 0.042. Under SHAP importance, *Epi.+Tran.* also receives a non-negligible value of 1.024, compared with 0.526 for *Len.*, although *Sem.* remains larger under this metric. The complete results in Appendix C.2, Table 5 show the same broad pattern, with variation in magnitude across trace-source and aggregation settings: episode-composition and transition features consistently contribute useful process-level information.

This importance pattern is consistent with the ablation results in Table 2. The full representation achieves the strongest performance, and removing any process feature family weakens the model. Together, the importance and ablation analyses suggest that Epi2Diff works by combining multiple forms of evidence rather than relying on a single proxy. Semantic features describe *what* the item is about, length features describe the *scale* of reasoning it elicits, episode-composition features describe *where* reasoning effort is allocated, and transition features describe *how* the reasoning process unfolds. This closes the loop with Finding 1: the same scale-allocation-flow structure that characterizes difficult items also provides a predictive signal for the final difficulty model.

5 Conclusion

We propose Epi2Diff, a framework that combines semantic item representations with episode-level features extracted from model-generated reasoning traces. By modeling reasoning scale, effort allocation, and local transitions alongside item content, the framework captures complementary signals of problem-solving burden. Across four benchmarks, it outperforms a broad range of classification and regression baselines. On SAT Math, higher difficulty is associated with more implementation-centered reasoning and stronger analysis-to-implementation transitions, rather than simply longer traces. These findings show that cognitively grounded abstractions of reasoning traces can provide scalable, interpretable proxy evidence for item difficulty modeling. Future work should test the robustness of these process signatures across additional assessment domains, reasoning models, and item formats.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024a. URL `https://arxiv.org/abs/2404.14219`.

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024b.

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 225–237, 2024.

Samah AlKhuzaey, Floriana Grasso, Terry R Payne, and Valentina Tamma. A systematic review of data-driven approaches to item difficulty prediction. In *International conference on artificial intelligence in education*, pp. 29–41. Springer, 2021.

Samah AlKhuzaey, Floriana Grasso, Terry R Payne, and Valentina Tamma. Text-based question difficulty prediction: A systematic review of automatic approaches. *International Journal of Artificial Intelligence in Education*, 34(3):862–914, 2024.

Luca Benedetto. A quantitative study of nlp approaches to question difficulty estimation. In *International conference on artificial intelligence in education*, pp. 428–434. Springer, 2023.

Maciej Besta, Julia Barth, Eric Schreiber, Ales Kubicek, Afonso Catarino, Robert Gerstenberger, Piotr Nyczyk, Patrick Iff, Yueling Li, Sam Houliston, et al. Reasoning language models: A blueprint. *arXiv preprint arXiv:2501.11223*, 2025.

Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025.

Han Chen, Ming Li, Chenguang Wang, Yijun Liang, Dawei Zhou, Hong Jiao, and Tianyi Zhou. LLMs struggle to measure what distinguishes students of different proficiency levels: A study of item discrimination in reading comprehension assessment. *arXiv preprint arXiv:2606.18709*, 2026.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators, 2020. URL https://arxiv.org/abs/2003.10555.

Ricardo Conejo, Eduardo Guzmán, Jose-Luis Perez-De-La-Cruz, and Beatriz Barros. An empirical study on the quantitative notion of task difficulty. *Expert Systems with Applications*, 41(2):594–606, 2014.

Christine DeMars. *Item response theory*. Oxford University Press, 2010.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Jiayu Ding, Shuming Ma, Lei Cui, Nanning Zheng, and Furu Wei. Longreasonarena: A long reasoning benchmark for large language models. *arXiv preprint arXiv:2508.19363*, 2025.

George Dueñas, Sergio Jimenez, and Geral Mateus Ferro. Upn-icc at bea 2024 shared task: Leveraging llms for multiple-choice questions difficulty prediction. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 542–550, 2024.

Daniel R Eignor. The standards for educational and psychological testing. 2013.

Chenrui Fan, Ming Li, Lichao Sun, and Tianyi Zhou. Missing premise exacerbates overthinking: Are reasoning models losing critical thinking skill? *arXiv preprint arXiv:2504.06514*, 2025.

Wanyong Feng, Peter Tran, Stephen Sireci, and Andrew S Lan. Reasoning and sampling-augmented mcq difficulty prediction via llms. In *International Conference on Artificial Intelligence in Education*, pp. 31–45. Springer, 2025a.

Yunzhen Feng, Julia Kempe, Cheng Zhang, Parag Jain, and Anthony Hartshorn. What characterizes effective reasoning? revisiting length, review, and structure of cot. *arXiv preprint arXiv:2509.19284*, 2025b.

Rishikesh Fulari and Jonathan Rusert. Utilizing machine learning to predict question difficulty and response time for enhanced test construction. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 528–533, 2024.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, and Danny Livshits. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Ronald K Hambleton, Hariharan Swaminathan, and H Jane Rogers. *Fundamentals of item response theory*, volume 2. Sage, 1991.

Peter Hase, Mohit Bansal, Peter Clark, and Sarah Wiegreffe. The unreasonable effectiveness of easy training data for hard tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7002–7024, 2024.

Jun He, Li Peng, Bo Sun, Lejun Yu, and Yinghui Zhang. Automatically predict question difficulty for reading comprehension exercises. In *2021 ieee 33rd international conference on tools with artificial intelligence (ictai)*, pp. 1398–1402. IEEE, 2021.

Fu-Yuan Hsu, Hahn-Ming Lee, Tao-Hsing Chang, and Yao-Ting Sung. Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Information Processing & Management*, 54(6):969–984, 2018.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Michael T Kane. Validating the interpretations and uses of test scores. *Journal of educational measurement*, 50(1):1–73, 2013.

Yeganeh Kordi, Nihal V Nayak, Max Zuo, Ilana Nguyen, and Stephen Bach. Revisiting generalization across difficulty levels: It's not so easy. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7014–7042, 2026.

Ming Li, Han Chen, Yunze Xiao, Jian Chen, Hong Jiao, and Tianyi Zhou. Can llms estimate student struggles? human-ai difficulty alignment with proficiency simulation for item difficulty prediction. *arXiv preprint arXiv:2512.18880*, 2025a.

Ming Li, Chenrui Fan, Yize Cheng, Soheil Feizi, and Tianyi Zhou. Schoenfeld's anatomy of mathematical reasoning by language models. *arXiv preprint arXiv:2512.19995*, 2025b.

Ming Li, Hong Jiao, Tianyi Zhou, Nan Zhang, Sydney Peters, and Robert W Lissitz. Item difficulty modeling using fine-tuned small and large language models. *Educational and Psychological Measurement*, 85(6):1065–1090, 2025c.

Ming Li, Nan Zhang, Chenrui Fan, Hong Jiao, Yanbin Fu, Sydney Peters, Qingshu Xu, Robert Lissitz, and Tianyi Zhou. Understanding the thinking process of reasoning models: A perspective from schoenfeld's episode theory. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 18278–18299, 2025d.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Anastassia Loukina, Su-Youn Yoon, Jennifer Sakano, Youhua Wei, and Kathy Sheehan. Textual complexity as a predictor of difficulty of listening items in language proficiency tests. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 3245–3253, 2016.

Arya D McCarthy, Kevin P Yancey, Geoffrey T LaFlair, Jesse Egbert, Manqian Liao, and Burr Settles. Jump-starting item parameters for adaptive language tests. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 883–899, 2021.

Samuel Messick. Validity of test interpretation and use. 1990.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Andrew Mullooly, Øistein Andersen, Luca Benedetto, Paula Buttery, Andrew Caines, Mark J. F. Gales, Yasin Karatay, Kate Knill, Adian Liusie, Vatsal Raina, and Shiva Taslimipoor. *The Cambridge Multiple-Choice Questions Reading Dataset*. 2023. doi: 10.17863/CAM.102185. URL https://www.repository.cam.ac.uk/handle/1810/358683.

Shadi Noroozi and Hossein Karami. A scrutiny of the relationship between cognitive load and difficulty estimates of language test items. *Language Testing in Asia*, 12(1):13, 2022.

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, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff,

Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

James R Parry. Ensuring fairness in difficulty and content among parallel assessments generated from a test-item database. *Online Submission*, 2020.

Kyle Perkins, Lalit Gupta, and Ravi Tammana. Predicting item difficulty in a reading comprehension test with an artificial neural network. *Language testing*, 12(1):34–53, 1995.

Sydney Peters, Nan Zhang, Hong Jiao, Ming Li, Tianyi Zhou, and Robert Lissitz. Text-based approaches to item difficulty modeling in large-scale assessments: A systematic review. *arXiv preprint arXiv:2509.23486*, 2025.

Pooya Razavi and Sonya J Powers. Estimating item difficulty using large language models and tree-based machine learning algorithms. *arXiv preprint arXiv:2504.08804*, 2025.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Ana-Cristina Rogoz and Radu Tudor Ionescu. Unibucllm: Harnessing llms for automated prediction of item difficulty and response time for multiple-choice questions. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 493–502, 2024.

Alan H Schoenfeld. *Mathematical problem solving*. Elsevier, 2014.

Alan H Schoenfeld. Learning to think mathematically: Problem solving, metacognition, and sense making in mathematics (reprint). *Journal of education*, 196(2):1–38, 2016.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, Alex Makelov, Alex Neitz, Alex Wei, Alexandra Barr, Alexandre Kirchmeyer, Alexey Ivanov, Alexi Christakis, Alistair Gillespie, Allison Tam, Ally Bennett, Alvin Wan, Alyssa Huang, Amy McDonald Sandjideh, Amy Yang, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrei Gheorghe, Andres Garcia Garcia, Andrew Braunstein, Andrew Liu, Andrew Schmidt, Andrey Mereskin, Andrey Mishchenko, Andy Applebaum, Andy Rogerson, Ann Rajan, Annie Wei, Anoop Kotha, Anubha Srivastava, Anushree Agrawal, Arun Vijayvergiya, Ashley Tyra, Ashvin Nair, Avi Nayak, Ben Eggers, Bessie Ji, Beth Hoover, Bill Chen, Blair Chen, Boaz Barak, Borys Minaiev, Botao Hao, Bowen Baker, Brad Lightcap, Brandon McKinzie, Brandon Wang, Brendan Quinn, Brian Fioca, Brian Hsu, Brian Yang, Brian Yu, Brian Zhang, Brittany Brenner, Callie Riggins Zetino, Cameron Raymond, Camillo Lugaresi, Carolina Paz, Cary Hudson, Cedric Whitney, Chak Li, Charles Chen, Charlotte Cole, Chelsea Voss, Chen Ding, Chen Shen, Chengdu Huang, Chris Colby, Chris Hallacy, Chris Koch, Chris Lu, Christina Kaplan, Christina Kim, CJ Minott-Henriques, Cliff Frey, Cody Yu, Coley Czarnecki, Colin Reid, Colin Wei, Cory Decareaux, Cristina Scheau, Cyril Zhang, Cyrus Forbes, Da Tang, Dakota Goldberg, Dan Roberts, Dana Palmie, Daniel Kappler, Daniel Levine, Daniel Wright, Dave Leo, David Lin, David Robinson, Declan Grabb, Derek Chen, Derek Lim, Derek Salama, Dibya Bhattacharjee, Dimitris Tsipras, Dinghua Li, Dingli Yu, DJ Strouse, Drew Williams, Dylan Hunn, Ed Bayes, Edwin Arbus, Ekin Akyurek, Elaine Ya Le, Elana Widmann, Eli Yani, Elizabeth Proehl, Enis Sert, Enoch Cheung, Eri Schwartz, Eric Han, Eric Jiang, Eric Mitchell, Eric Sigler, Eric Wallace, Erik Ritter, Erin Kavanaugh, Evan Mays, Evgenii Nikishin, Fangyuan Li, Felipe Petroski Such, Filipe de Avila Belbute Peres, Filippo Raso, Florent Bekerman, Foivos Tsimpourlas, Fotis Chantzis, Francis Song, Francis Zhang, Gaby Raila, Garrett McGrath, Gary Briggs, Gary Yang, Giambattista Parascandolo, Gildas Chabot, Grace Kim, Grace Zhao, Gregory Valiant, Guillaume Leclerc, Hadi Salman,

Hanson Wang, Hao Sheng, Haoming Jiang, Haoyu Wang, Haozhun Jin, Harshit Sikchi, Heather Schmidt, Henry Aspegren, Honglin Chen, Huida Qiu, Hunter Lightman, Ian Covert, Ian Kivlichan, Ian Silber, Ian Sohl, Ibrahim Hammoud, Ignasi Clavera, Ikai Lan, Ilge Akkaya, Ilya Kostrikov, Irina Kofman, Isak Etinger, Ishaan Singal, Jackie Hehir, Jacob Huh, Jacqueline Pan, Jake Wilczynski, Jakub Pachocki, James Lee, James Quinn, Jamie Kiros, Janvi Kalra, Jasmyn Samaroo, Jason Wang, Jason Wolfe, Jay Chen, Jay Wang, Jean Harb, Jeffrey Han, Jeffrey Wang, Jennifer Zhao, Jeremy Chen, Jerene Yang, Jerry Tworek, Jesse Chand, Jessica Landon, Jessica Liang, Ji Lin, Jiancheng Liu, Jianfeng Wang, Jie Tang, Jihan Yin, Joanne Jang, Joel Morris, Joey Flynn, Johannes Ferstad, Johannes Heidecke, John Fishbein, John Hallman, Jonah Grant, Jonathan Chien, Jonathan Gordon, Jongsoo Park, Jordan Liss, Jos Kraaijeveld, Joseph Guay, Joseph Mo, Josh Lawson, Josh McGrath, Joshua Vendrow, Joy Jiao, Julian Lee, Julie Steele, Julie Wang, Junhua Mao, Kai Chen, Kai Hayashi, Kai Xiao, Kamyar Salahi, Kan Wu, Karan Sekhri, Karan Sharma, Karan Singhal, Karen Li, Kenny Nguyen, Keren Gu-Lemberg, Kevin King, Kevin Liu, Kevin Stone, Kevin Yu, Kristen Ying, Kristian Georgiev, Kristie Lim, Kushal Tirumala, Kyle Miller, Lama Ahmad, Larry Lv, Laura Clare, Laurance Fauconnet, Lauren Itow, Lauren Yang, Laurentia Romaniuk, Leah Anise, Lee Byron, Leher Pathak, Leon Maksin, Leyan Lo, Leyton Ho, Li Jing, Liang Wu, Liang Xiong, Lien Mamitsuka, Lin Yang, Lindsay McCallum, Lindsey Held, Liz Bourgeois, Logan Engstrom, Lorenz Kuhn, Louis Feuvrier, Lu Zhang, Lucas Switzer, Lukas Kondraciuk, Lukasz Kaiser, Manas Joglekar, Mandeep Singh, Mandip Shah, Manuka Stratta, Marcus Williams, Mark Chen, Mark Sun, Marselus Cayton, Martin Li, Marvin Zhang, Marwan Aljubeh, Matt Nichols, Matthew Haines, Max Schwarzer, Mayank Gupta, Meghan Shah, Melody Huang, Meng Dong, Mengqing Wang, Mia Glaese, Micah Carroll, Michael Lampe, Michael Malek, Michael Sharman, Michael Zhang, Michele Wang, Michelle Pokrass, Mihai Florian, Mikhail Pavlov, Miles Wang, Ming Chen, Mingxuan Wang, Minnia Feng, Mo Bavarian, Molly Lin, Moose Abdool, Mostafa Rohaninejad, Nacho Soto, Natalie Staudacher, Natan LaFontaine, Nathan Marwell, Nelson Liu, Nick Preston, Nick Turley, Nicklas Ansman, Nicole Blades, Nikil Pancha, Nikita Mikhaylin, Niko Felix, Nikunj Handa, Nishant Rai, Nitish Keskar, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Oona Gleeson, Pamela Mishkin, Patryk Lesiewicz, Paul Baltescu, Pavel Belov, Peter Zhokhov, Philip Pronin, Phillip Guo, Phoebe Thacker, Qi Liu, Qiming Yuan, Qinghua Liu, Rachel Dias, Rachel Puckett, Rahul Arora, Ravi Teja Mullapudi, Raz Gaon, Reah Miyara, Rennie Song, Rishabh Aggarwal, RJ Marsan, Robel Yemiru, Robert Xiong, Rohan Kshirsagar, Rohan Nuttall, Roman Tsiupa, Ronen Eldan, Rose Wang, Roshan James, Roy Ziv, Rui Shu, Ruslan Nigmatullin, Saachi Jain, Saam Talaie, Sam Altman, Sam Arnesen, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Sarah Yoo, Savannah Heon, Scott Ethersmith, Sean Grove, Sean Taylor, Sebastien Bubeck, Sever Banesiu, Shaokyi Amdo, Shengjia Zhao, Sherwin Wu, Shibani Santurkar, Shiyu Zhao, Shraman Ray Chaudhuri, Shreyas Krishnaswamy, Shuaiqi, Xia, Shuyang Cheng, Shyamal Anadkat, Simón Posada Fishman, Simon Tobin, Siyuan Fu, Somay Jain, Song Mei, Sonya Egoian, Spencer Kim, Spug Golden, SQ Mah, Steph Lin, Stephen Imm, Steve Sharpe, Steve Yadlowsky, Sulman Choudhry, Sungwon Eum, Suvansh Sanjeev, Tabarak Khan, Tal Stramer, Tao Wang, Tao Xin, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Degry, Thomas Shadwell, Tianfu Fu, Tianshi Gao, Timur Garipov, Tina Sriskandarajah, Toki Sherbakov, Tomer Kaftan, Tomo Hiratsuka, Tongzhou Wang, Tony Song, Tony Zhao, Troy Peterson, Val Kharitonov, Victoria Chernova, Vineet Kosaraju, Vishal Kuo, Vitchyr Pong, Vivek Verma, Vlad Petrov, Wanning Jiang, Weixing Zhang, Wenda Zhou, Wenlei Xie, Wenting Zhan, Wes McCabe, Will DePue, Will Ellsworth, Wulfie Bain, Wyatt Thompson, Xiangning Chen, Xiangyu Qi, Xin Xiang, Xinwei Shi, Yann Dubois, Yaodong Yu, Yara Khakbaz, Yifan Wu, Yilei Qian, Yin Tat Lee, Yinbo Chen, Yizhen Zhang, Yizhong Xiong, Yonglong Tian, Young Cha, Yu Bai, Yu Yang, Yuan Yuan, Yuanzhi Li, Yufeng Zhang, Yuguang Yang, Yujia Jin, Yun Jiang, Yunyun Wang, Yushi Wang, Yutian Liu, Zach Stubenvoll, Zehao Dou, Zheng Wu, and Zhigang Wang. Openai gpt-5 system card, 2025. URL https://arxiv.org/abs/2601.03267.

John Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive science*, 12(2):257–285, 1988.

John Sweller. Cognitive load theory. In *Psychology of learning and motivation*, volume 55, pp. 37–76. Elsevier, 2011.

Anaïs Tack, Siem Buseyne, Changsheng Chen, Robbe D'hondt, Michiel De Vrindt, Alireza Gharahighehi, Sameh Metwaly, Felipe Kenji Nakano, and Ann-Sophie Noreillie. Itec at bea 2024 shared task: Predicting difficulty and response time of medical exam questions with statistical, machine learning, and language models. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 512–521, 2024.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

Hariram Veeramani, Surendrabikram Thapa, Natarajan Balaji Shankar, and Abeer Alwan. Large language model-based pipeline for item difficulty and response time estimation for educational assessments. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 561–566, 2024.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, 2024. URL https://arxiv.org/abs/2412.13663.

Kelly Wauters, Piet Desmet, and Wim Van Den Noortgate. Item difficulty estimation: An auspicious collaboration between data and judgment. *Computers & Education*, 58(4): 1183–1193, 2012.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wei Xiong, Hanning Zhang, Chenlu Ye, Lichang Chen, Nan Jiang, and Tong Zhang. Self-rewarding correction for mathematical reasoning. *arXiv preprint arXiv:2502.19613*, 2025.

Kang Xue, Victoria Yaneva, Christopher Runyon, and Peter Baldwin. Predicting the difficulty and response time of multiple choice questions using transfer learning. In *Proceedings of the fifteenth workshop on innovative use of NLP for building educational applications*, pp. 193–197, 2020.

Victoria Yaneva, Peter Baldwin, Janet Mee, et al. Predicting the difficulty of multiple choice questions in a high-stakes medical exam. In *Proceedings of the fourteenth workshop on innovative use of NLP for building educational applications*, pp. 11–20, 2019.

Victoria Yaneva, Kai North, Peter Baldwin, Le An Ha, Saed Rezayi, Yiyun Zhou, Sagnik Ray Choudhury, Polina Harik, and Brian Clauser. Findings from the first shared task on automated prediction of difficulty and response time for multiple-choice questions. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 470–482, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhihao Yuan, Yunze Xiao, Ming Li, Weihao Xuan, Richard Tong, Mona Diab, and Tom Mitchell. Towards valid student simulation with large language models. *arXiv preprint arXiv:2601.05473*, 2026.

Leonidas Zotos, Hedderik van Rijn, and Malvina Nissim. Are you doubtful? oh, it might be difficult then! exploring the use of model uncertainty for question difficulty estimation. *arXiv preprint arXiv:2412.11831*, 2024.

A Experiment Setup

A.1 Datasets

We evaluate on four educational benchmarks spanning medical, English reading-comprehension, verbal reasoning, and mathematical reasoning domains. **(1) USMLE.** The USMLE dataset (Yaneva et al., 2024) contains 667 multiple-choice items from a high-stakes medical assessment setting, with difficulty labels derived from field-test data collected from more than 300 medical students per item. Following the original setup, we use the provided linearly transformed proportion-correct scores in $[0, 1.3]$ as continuous difficulty labels, where higher values indicate more difficult items. **(2) Cambridge.** The Cambridge dataset (Mullooly et al., 2023) contains 120 reading passages and 793 multiple-choice questions for English reading comprehension, with rescaled IRT b-parameters in $[0, 100]$ as continuous difficulty labels. **(3) SAT Reading & Writing.** This dataset is constructed from the SAT Question Bank[1] after excluding figure-dependent items. It contains 1,338 verbal reasoning questions covering reading comprehension, writing, and language conventions, with three discrete labels: Easy, Medium, and Hard. **(4) SAT Math.** This dataset is also collected from the SAT Question Bank and contains 1,075 mathematics questions covering algebra, geometry, data analysis, and multi-step problem solving, using the same Easy, Medium, and Hard labels. Detailed train–test split sizes and protocols are provided in Appendix A.4.

A.2 Models and Baselines

Our approach begins by leveraging two strong reasoning-oriented LLMs, **QwQ-32B** (Team, 2025) and **Qwen3-32B** (Yang et al., 2025), in reasoning mode, to produce explicit reasoning traces for each item. These generated traces serve as intermediate supervision signals and are subsequently used as the foundation for downstream difficulty prediction.

To provide a fair and comprehensive comparison against mainstream item difficulty prediction methods, we evaluate baselines from **five** representative paradigms, spanning both conventional small-model approaches and recent LLM-based strategies. **(1) SLM fine-tuning:** We fine-tune encoder-based small language models, including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ModernBERT (Warner et al., 2024), and ELECTRA (Clark et al., 2020), as standard supervised baselines for difficulty prediction. **(2) LLM zero-shot inference:** we directly prompt frontier LLMs, including GPT-4o (OpenAI et al., 2024), GPT-5 (Singh et al., 2025), QwQ-32B (Team, 2025), and Qwen3-32B (Yang et al., 2025), to predict item difficulty without task-specific training. **(3) LLM in-context learning:** we further evaluate the same set of LLMs under few-shot prompting with different shot settings, in order to test whether exemplar-based conditioning improves prediction quality, including GPT-4o (OpenAI et al., 2024), GPT-5 (Singh et al., 2025), QwQ-32B (Team, 2025), and Qwen3-32B (Yang et al., 2025). **(4) LLM supervised fine-tuning:** we conduct full-parameter supervised fine-tuning on a group of compact open-source decoder models, including Qwen2.5-3B (Yang et al., 2024), Qwen3-4B (Yang et al., 2025), Phi-3.5-mini (Abdin et al., 2024a), Phi-4-mini (Abdin et al., 2024b), and Llama-3.2-3B (Grattafiori et al., 2024). **(5) Parameter-efficient fine-tuning:** on the same model set, we additionally implement **LoRA**-based adaptation as a more efficient alternative to full fine-tuning (Hu et al., 2022), using a fixed rank of $r = 16$ for all experiments. Together, these baselines cover a broad methodological spectrum, allowing us to compare our approach against both classical supervised models and modern prompting- and fine-tuning-based LLM methods under a unified experimental setup.

[1] https://satsuitequestionbank.collegeboard.org/

A.3 Evaluation Metrics

For classification tasks, we evaluate performance using accuracy and macro-F1. Given ground-truth labels $\{y_i\}_{i=1}^N$ and predictions $\{\hat{y}_i\}_{i=1}^N$, accuracy is defined as

$$\text{ACC} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\hat{y}_i = y_i), \tag{5}$$

where $\mathbf{1}(\cdot)$ is the indicator function.

Macro-F1 is defined as the unweighted average of class-wise F1 scores:

$$\text{Macro-F1} = \frac{1}{C}\sum_{c=1}^{C} F1_c, \tag{6}$$

where C is the number of classes. For each class c, the F1 score is computed as

$$F1_c = \frac{2P_cR_c}{P_c + R_c}, \tag{7}$$

where P_c and R_c denote the precision and recall for class c, respectively:

$$P_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c}, \qquad R_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c}. \tag{8}$$

For regression tasks, we report RMSE and R^2. Given ground-truth values $\{y_i\}_{i=1}^N$, predictions $\{\hat{y}_i\}_{i=1}^N$, and the mean target value

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i, \tag{9}$$

RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \tag{10}$$

and the coefficient of determination is defined as

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}. \tag{11}$$

A lower RMSE indicates smaller prediction error, while a higher R^2 indicates better fit to the ground-truth difficulty values.

A.4 Dataset Splits

For Cambridge, SAT Math, and SAT Reading & Writing, we use fixed stratified 80/20 train–test splits with random seed 42. Table 3 reports the resulting number of items in each partition. For USMLE, we use the official train–test split provided with the dataset rather than creating a new split.

A.5 Feature-Group Importance

We quantify the contribution of each feature group using both group-wise permutation importance and SHAP importance on the held-out test set. Let f denote the fitted predictor, let $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$ denote the test set, and let G be a group of feature columns. For SAT Math, the evaluation score $\mathcal{M}$ is macro-F1. We jointly permute all columns in G using the same random permutation π_G, which preserves dependencies among features within the group. The permutation importance of G is the resulting decrease in test performance:

$$I_{\text{perm}}(G) = \mathcal{M}(f, \mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}) - \mathcal{M}\left(f, \mathbf{X}_{\text{test}}^{\pi_G}, \mathbf{y}_{\text{test}}\right). \tag{12}$$

Dataset	Train items	Test items	Total
Cambridge	634	159	793
SAT Math	860	215	1,075
SAT Reading & Writing	1,070	268	1,338

Table 3: Numbers of items in the fixed train–test splits. Cambridge, SAT Math, and SAT Reading & Writing use stratified 80/20 splits with seed 42. USMLE uses its official train–test split and is therefore not included in the table.

Thus, a larger value indicates that shuffling the group causes a larger degradation in predictive performance.

For SHAP importance, let $\phi_{ij}^{(c)}$ be the SHAP value of feature j for test item i and class c. For a feature group G, we report the sum of mean absolute SHAP values over its constituent features:

$$I_{\text{SHAP}}(G) = \sum_{j \in G} \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \left(\frac{1}{C} \sum_{c=1}^{C} \left| \phi_{ij}^{(c)} \right| \right), \tag{13}$$

where C is the number of classes. This class average is used for the multiclass SAT Math classifier; for a scalar-output model, the inner class average reduces to $|\phi_{ij}|$. We apply these definitions to the semantic (*Sem.*), length (*Len.*), episode-composition (*Epi.*), transition (*Tran.*), and combined episode-plus-transition (*Epi.+Tran.*) groups reported in Section 4 and Appendix C.2.

B Related Work

B.1 Item Difficulty Prediction

Estimating item difficulty has traditionally relied on response-data calibration under CTT or IRT, which remains central for calibrated assessment but is costly for newly authored items because it typically requires field testing before items can be operationalized (DeMars, 2010; Hsu et al., 2018; AlKhuzaey et al., 2024; Peters et al., 2025). Expert judgment has also been considered as a lower-cost alternative, but prior work reports limited or variable agreement between expert ratings and empirical psychometric difficulty estimates, limiting its reliability as a scalable substitute (Conejo et al., 2014; Wauters et al., 2012). To reduce this reliance on response data, early NLP work treated item difficulty prediction as a text-based supervised learning problem. Early approaches also used artificial neural networks with linguistic, propositional, and cognitive-process features for reading-comprehension items (Perkins et al., 1995), while later work extracted over 100 hand-crafted textual-complexity features, such as word length, syntactic complexity, and readability indices, to predict difficulty in language assessment (Loukina et al., 2016). Subsequent work incorporated richer semantic representations and item-component similarities, including Word2Vec-based features (Hsu et al., 2018), and showed that NLP-derived linguistic and syntactic features are predictive for high-stakes medical multiple-choice questions (Yaneva et al., 2019). Neural models further extended this line, including CNN-based models for reading-comprehension difficulty prediction (He et al., 2021). Overall, this literature has gradually shifted from hand-crafted linguistic and psychometric proxies toward learned textual representations (AlKhuzaey et al., 2024; Peters et al., 2025).

More recently, transformer-based representations have become strong baselines for item difficulty prediction. BERT features have been used in a multi-task generalized linear model to jump-start item-parameter estimates for adaptive language testing (McCarthy et al., 2021), and Transformer-based models have been shown to outperform traditional feature-based, TF-IDF, and Word2Vec baselines for question difficulty estimation (Benedetto, 2023; Devlin et al., 2019). Other work combines statistical, machine-learning, and language-model approaches, while also showing that simpler feature-based models can remain competitive in some settings (Tack et al., 2024). Most recently, comparisons across a broad family of SLM- and LLM-based approaches suggest that fine-tuned encoder models such as BERT and RoBERTa can outperform larger but less task-specialized LLM alternatives on a medical assessment benchmark (Li et al., 2025c; Liu et al., 2019). In conclusion, these studies establish strong text-representation baselines for item difficulty prediction, but they largely model difficulty from item text, metadata, annotations, embeddings, or other item-level representations, leaving process-level solving burden less directly explored.

B.2 LLM-Based Methods for Item Difficulty Prediction

With the emergence of LLMs, recent work has explored using model outputs and behavior as additional signals for item difficulty prediction. One line of work prompts LLMs to answer items, simulate examinee-like responses, generate rationales, or directly estimate difficulty from item content. Zero-shot LLM-generated answers and answer variability have been used to support multiple-choice item difficulty and response-time prediction (Rogoz & Ionescu, 2024). Other work simulates medical-candidate responses across prompting scenarios and extracts features from these responses for downstream prediction (Dueñas et al., 2024). More broadly, strong item-solving ability does not necessarily translate into accurate estimates of human difficulty, suggesting that direct LLM judgments may reflect model-centric difficulty rather than empirical human difficulty (Li et al., 2025a). Similarly, LLMs show only limited alignment with human-calibrated item discrimination, indicating that difficulty is not the only psychometric property that remains challenging for LLM-based assessment (Chen et al., 2026). Another line of work uses LLMs less as end-to-end judges and more as generators of auxiliary predictive signals. These signals include LLM-based pipelines using semantic and linguistic features such as named entities and semantic roles (Veeramani et al., 2024), reasoning- and sampling-based features (Feng et al., 2025a), uncertainty-aware features (Zotos et al., 2024), direct ratings and LLM-extracted cognitive

or linguistic features combined with tree-based machine-learning models (Razavi & Powers, 2025), and fine-tuned SLM/LLM representations for item difficulty modeling (Li et al., 2025c). These studies indicate that LLM outputs can provide useful auxiliary cues for item difficulty prediction, but they do not uniformly outperform strong specialized baselines, and their extracted signals are usually treated as predictive features rather than structured, interpretable evidence of solving processes aligned with human psychometric difficulty.

A related but distinct body of work studies difficulty in broader LLM learning and evaluation settings. Prior studies examine how difficulty shapes easy-to-hard generalization, cross-difficulty transfer, and benchmark design where difficulty is scaled by required reasoning length (Hase et al., 2024; Kordi et al., 2026; Ding et al., 2025). This work is relevant because it shows that difficulty affects model behavior and evaluation, though it does not focus on empirically calibrated human item difficulty in educational assessment.

B.3 Reasoning Traces as Process Signals

A parallel line of research asks not whether LLMs can directly judge difficulty, but whether generated reasoning traces contain structured information about how models process problems. Chain-of-thought prompting made intermediate reasoning steps explicit (Wei et al., 2022), and newer reasoning-oriented models further amplified this trend by producing longer, human-readable traces that may expose self-reflection, verification, and strategy adaptation during inference (Guo et al., 2025). Recent work on mathematical reasoning and reasoning language models further situates such traces within broader efforts to analyze model reasoning behavior and organization (Ahn et al., 2024; Besta et al., 2025). This suggests that reasoning traces may reflect model-side process structure beyond verbosity, although they should not be interpreted as direct observations of human cognition.

Recent work has started to operationalize this idea through more explicit structural abstractions. For example, Schoenfeld's Episode Theory has been applied to reasoning-model traces, showing that model-generated solutions can be decomposed into interpretable functional episodes (Li et al., 2025d). More generally, studies on reasoning efficiency and trace quality suggest that effective reasoning cannot be reduced to response length alone; rather, it depends on how models allocate effort across exploration, execution, review, and verification, and longer traces may contain redundant or inefficient reasoning (Chen et al., 2024; Feng et al., 2025b; Bogdan et al., 2025; Aggarwal & Welleck, 2025; Fan et al., 2025). Still, most prior trace-based work has focused on interpretability, behavioral diagnosis, or reasoning-efficiency analysis, rather than using reasoning traces as structured predictive signals for downstream human item difficulty estimation. This leaves open the central question of our work: whether model-generated reasoning traces can be transformed into scalable and interpretable process-level signals for estimating the human difficulty of unseen items.

C Supplemental Ablation Study

C.1 Transition Feature Design

We explored several alternative parameterizations of transition features rather than assuming from the outset that adjacent bigram counts would be the best choice. Specifically, in addition to the default raw bigram transition counts, we considered globally normalized bigram ratios, unique bigram counts, raw trigram counts, and deduplicated trigram counts computed after collapsing consecutive repeated episode labels, inspired by ThinkARM (Li et al., 2025b). Empirically, however, none of these alternatives produced more consistent gains than the raw bigram count representation across our SAT Math settings. A plausible explanation is that adjacent bigram counts already capture the dominant local reasoning flow that is most relevant for difficulty prediction, while the alternative designs either discard useful magnitude information (e.g., ratio or unique-count variants) or move to a more specific and sparser representation (e.g., trigram variants).

	QwQ		Qwen		Both		QwQ (Role)		Qwen (Role)		Both (Role)	
Method	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**
Base + 2-gram Cnt.	0.684	0.681	0.712	0.708	0.707	0.708	0.698	0.697	0.712	0.712	**0.730**	**0.728**
Base + 2-gram Rat.	0.670	0.667	0.684	0.679	0.674	0.670	0.688	0.687	0.702	0.700	0.688	0.690
Base + 2-gram Unq.	0.670	0.671	0.693	0.688	0.688	0.688	0.674	0.677	0.684	0.683	0.688	0.691
Base + 3-gram Cnt.	0.651	0.651	0.684	0.679	0.679	0.680	0.670	0.669	0.681	0.681	0.698	0.698
Base + 3-gram Dedup.	0.642	0.639	0.679	0.673	0.684	0.681	0.688	0.690	0.698	0.699	0.684	0.684
Base + 2-gram Cnt. + 2-gram Rat.	0.660	0.658	0.693	0.687	0.679	0.675	0.660	0.662	0.702	0.701	0.688	0.690
Base + 2-gram Cnt. + 2-gram Unq.	0.651	0.649	0.684	0.679	0.674	0.669	0.674	0.673	0.702	0.703	0.721	0.721
Base + 2-gram Cnt. + 3-gram Cnt.	0.647	0.645	0.698	0.693	0.698	0.697	0.679	0.680	0.679	0.682	0.688	0.688
Base + 2-gram Cnt. + 3-gram Dedup.	0.651	0.646	0.665	0.656	0.665	0.660	0.679	0.679	0.698	0.699	0.684	0.686

Table 4: **Transition-design ablation on SAT Math across reasoning-trace sources and aggregation settings.** *Base* denotes the semantic representation combined with the non-transition process features, i.e., length and episode-composition features. *Cnt.*, *Rat.*, and *Unq.* denote count, global ratio, and unique-count variants, respectively. *Dedup.* denotes trigram transitions computed after collapsing consecutive repeated episode labels. Across all settings, the default *2-gram Cnt.* transition design performs best, while alternative parameterizations and concatenated variants do not yield consistent gains.

We further tested whether alternative transition parameterizations could provide complementary information on top of the default bigram count features by concatenating them with the base representation. Some of these combinations remained competitive, but none consistently surpassed the default bigram count design. We therefore do not adopt mixed transition representations in the final model. One likely reason is that these variants are not sufficiently independent from the adjacent bigram counts: higher-order or reformulated transitions partially re-encode the same local-order signal, so concatenation mainly increases dimensionality, sparsity, and redundancy. This issue is especially pronounced for trigram features: with eight episode labels, the transition space expands from $8^2 = 64$ bigram features to $8^3 = 512$ trigram features, making the representation much sparser and more prone to amplifying noise under our relatively limited data regime. Taken together, these results suggest that raw adjacent bigram counts provide the best trade-off among information retention, robustness, compactness, and interpretability, which is why we use them as the default transition representation.

C.2 Additional Feature-Importance and Directional-Effect Analyses

This appendix provides additional analyses supporting the interpretation of SAT Math difficulty as structured problem-solving burden. Table 5 reports feature-group importance across reasoning-trace settings, while Table 6 reports fine-grained standardized average marginal effects (SAMEs) for individual process features under the profile-aggregated *Both (Role)* setting. Together, the two tables separate two complementary questions: whether each feature group contributes to prediction, and how individual process features directionally relate to the fitted difficulty score.

Table 5 shows that episode-induced process features provide predictive signal across settings. Although semantic features remain a strong anchor when they are included, the combined episode-composition and transition group (*Epi.+Tran.*) consistently contributes nonzero importance. In several permutation-importance settings, *Epi.+Tran.* is comparable to or larger than the semantic feature group. For example, under Both (Role), *Epi.+Tran.* reaches 0.136 in permutation importance, exceeding *Sem.* at 0.084 and *Len.* at 0.042. The SHAP results lead to a compatible conclusion: under Both (Role), *Epi.+Tran.* has a mean absolute SHAP importance of 1.024, larger than *Len.* at 0.526. These results suggest that the predictor uses not only item semantics and reasoning length, but also the organization of reasoning across episode types and transitions.

Table 6 further explains the direction of these process signals. Among length features, total trace length and *think* length have the largest positive effects, whereas final *answer* length has a smaller positive effect. This indicates that higher predicted difficulty is associated mainly with additional intermediate reasoning rather than merely longer final responses.

Feature	QwQ	Qwen	QwQ$^+$	Qwen$^+$	Both	QwQ (Role)	Qwen (Role)	Both (Role)
Permutation Importance								
Sem.	–	–	0.130	0.058	0.095	0.116	0.103	0.084
Len.	0.055	0.019	0.062	0.027	0.052	0.046	0.045	0.042
Epi.	0.130	0.225	0.096	0.089	0.069	0.057	0.058	0.024
Tran.	0.018	0.015	0.011	0.031	0.069	0.027	0.067	0.064
Epi.+Tran.	0.141	0.184	0.133	0.140	0.088	0.083	0.166	0.136
SHAP Importance								
Sem.	–	–	3.380	2.420	4.683	4.175	3.775	4.273
Len.	0.406	0.231	0.264	0.252	0.450	0.439	0.373	0.526
Epi.	1.378	0.545	0.542	0.437	0.635	0.598	0.366	0.436
Tran.	1.470	0.450	0.200	0.206	0.425	0.466	0.570	0.588
Epi.+Tran.	2.848	0.995	0.742	0.643	1.060	1.065	0.936	1.024

Table 5: **Feature importance on SAT Math across reasoning-trace settings.** *Sem.*, *Len.*, *Epi.*, and *Tran.* denote the semantic, length, episode-composition, and transition features, respectively. *Epi.+Tran.* denotes the combined episode-composition and transition feature group. *QwQ* and *Qwen* denote structure-only settings without semantic features, while the superscript $+$ in *QwQ*$^+$ and *Qwen*$^+$ indicates the corresponding full-feature settings that include semantic features. *Both* denotes the combined QwQ and Qwen trace-source setting, and *Role* denotes averaging across direct and proficiency-conditioned weak, medium, and strong model profiles. Permutation importance is computed by shuffling each feature as a group, and SHAP importance is reported as the mean absolute SHAP value aggregated at the level of *Sem.*, *Len.*, *Epi.*, *Tran.*, and *Epi.+Tran.* Semantic importance is not applicable to the structure-only settings *QwQ* and *Qwen*, so those entries are marked as "–".

At the episode level, *Implement* has the largest positive count effect, and the *Implement* ratio is also positive. In contrast, the ratios of *Verify*, *Read*, and *Plan* are negative. This pattern suggests that difficulty is not simply associated with uniformly longer traces; instead, the reasoning budget shifts toward concrete solution execution.

The transition features provide a local sequential view of this execution-centered burden. The strongest positive transition effect is *Analyze→Implement*, followed by *Implement→Implement* and *Implement→Analyze*. Positive effects for *Plan→Implement*, *Analyze→Analyze*, and *Verify→Implement* further indicate that harder items tend to involve movement from analysis or planning into execution, sustained execution, and local refinement around implementation. Therefore, the process signal captured by the episode-induced representation is not reducible to token count alone. Length captures the scale of the induced burden, episode features capture how reasoning effort is allocated, and transition features capture how the reasoning process unfolds over time.

C.3 Additional Ablation Results

To provide a more comprehensive comparison with Epi2Diff, we further conduct several additional experiments with alternative feature configurations, as shown in Table 7. These comparisons include using only LLM-extracted item-text features, combining item semantic representations with LLM-extracted features, and using only embeddings derived from the reasoning trace. Overall, the results show that Epi2Diff remains competitive against these alternatives. The LLM-extracted features provide useful item-level signals, but relying on them alone or combining them with item semantics does not consistently improve over the proposed fine-grained feature design. In addition, the trace-only setting performs less favorably, suggesting that reasoning traces by themselves may not capture sufficient information for robust prediction. These findings further support the effectiveness of explicitly modeling fine-grained item features in Epi2Diff.

Length		Episode		Transition			
Feature label	**Std. AME**	**Feature label**	**Std. AME**	**Feature label**	**Std. AME**	**Feature label**	**Std. AME**
Length (Total)	0.2374	Count (Imp)	0.0875	Trans (Ana → Imp)	0.0917	Trans (Mon → Ver)	0.0000
Length (Think)	0.1509	Count (Ana)	0.0417	Trans (Imp → Imp)	0.0729	Trans (Ana → Exp)	0.0000
Length (Answer)	0.0383	Ratio (Imp)	0.0086	Trans (Imp → Ana)	0.0206	Trans (Ans → Ana)	0.0000
		Count (Plan)	0.0073	Trans (Ans → Ans)	0.0105	Trans (Read → Ans)	0.0000
		Count (Mon)	0.0053	Trans (Plan → Ana)	0.0074	Trans (Exp → Ans)	0.0000
		Count (Ans)	0.0038	Trans (Ana → Ana)	0.0069	Trans (Ana → Ver)	0.0000
		Ratio (Ana)	0.0024	Trans (Imp → Plan)	0.0059	Trans (Ver → Read)	0.0000
		Count (Read)	0.0000	Trans (Plan → Imp)	0.0054	Trans (Exp → Imp)	0.0000
		Count (Exp)	-0.0003	Trans (Ver → Imp)	0.0052	Trans (Read → Exp)	0.0000
		Ratio (Ans)	-0.0012	Trans (Imp → Read)	0.0045	Trans (Read → Imp)	0.0000
		Ratio (Plan)	-0.0013	Trans (Mon → Ana)	0.0038	Trans (Ver → Plan)	-0.0001
		Ratio (Read)	-0.0015	Trans (Ana → Plan)	0.0037	Trans (Plan → Plan)	-0.0001
		Count (Ver)	-0.0017	Trans (Mon → Mon)	0.0019	Trans (Ans → Mon)	-0.0001
		Ratio (Exp)	-0.0030	Trans (Ans → Ver)	0.0017	Trans (Ver → Ana)	-0.0001
		Ratio (Mon)	-0.0192	Trans (Ans → Read)	0.0016	Trans (Mon → Read)	-0.0001
		Ratio (Ver)	-0.0278	Trans (Imp → Ans)	0.0015	Trans (Ana → Mon)	-0.0001
				Trans (Read → Mon)	0.0011	Trans (Read → Read)	-0.0001
				Trans (Imp → Mon)	0.0009	Trans (Exp → Read)	-0.0001
				Trans (Mon → Imp)	0.0008	Trans (Ver → Ver)	-0.0002
				Trans (Ana → Read)	0.0005	Trans (Read → Plan)	-0.0003
				Trans (Imp → Exp)	0.0004	Trans (Imp → Ver)	-0.0003
				Trans (Plan → Exp)	0.0004	Trans (Ans → Plan)	-0.0003
				Trans (Plan → Read)	0.0004	Trans (Ans → Exp)	-0.0003
				Trans (Plan → Ver)	0.0004	Trans (Exp → Plan)	-0.0003
				Trans (Ver → Ans)	0.0004	Trans (Exp → Ver)	-0.0004
				Trans (Mon → Ans)	0.0003	Trans (Exp → Mon)	-0.0004
				Trans (Mon → Exp)	0.0003	Trans (Exp → Exp)	-0.0005
				Trans (Plan → Mon)	0.0003	Trans (Ana → Ans)	-0.0007
				Trans (Read → Ana)	0.0003	Trans (Ver → Mon)	-0.0010
				Trans (Ans → Imp)	0.0002	Trans (Plan → Ans)	-0.0010
				Trans (Exp → Ana)	0.0002	Trans (Ver → Exp)	-0.0017
				Trans (Mon → Plan)	0.0001	Trans (Read → Ver)	-0.0029

Table 6: **Fine-grained standardized directional effects on SAT Math difficulty.** Values are standardized average marginal effects (SAME) on the expected ordinal difficulty score of the fitted XGBoost classifier, $\hat{s}(x) = \hat{p}_{\text{medium}}(x) + 2\hat{p}_{\text{hard}}(x)$, where easy $= 0$, medium $= 1$, and hard $= 2$. Positive values indicate that increasing the corresponding feature by one empirical training-set standard deviation shifts the predicted distribution toward harder classes, whereas negative values indicate shifts toward easier classes. The magnitude corresponds to the average change in expected difficulty score on the ordered three-class scale.

	QwQ		Qwen		Both		QwQ (Role)		Qwen (Role)		Both (Role)	
Method	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**
Sem. + FFea.	**0.684**	**0.681**	**0.712**	**0.708**	**0.707**	**0.708**	**0.698**	**0.697**	**0.712**	**0.712**	**0.730**	**0.728**
LLM Fea. Only	0.674	0.673	0.674	0.674	0.669	0.668	0.674	0.674	0.688	0.687	0.683	0.685
Item Sem. + LLM Fea.	0.655	0.654	0.655	0.653	0.660	0.658	0.669	0.666	0.678	0.676	0.655	0.652
Trace Emb. Only	0.647	0.642	0.605	0.597	0.628	0.623	0.628	0.622	0.637	0.632	0.656	0.653

Table 7: Additional comparison results across alternative feature configurations. *Sem. + FFea* denotes the original setting that combines item semantic representations with the proposed fine-grained features. *LLM Fea. Only* uses only the features extracted by the LLM from the item text itself, following the LLM-based feature extraction strategy of Razavi & Powers (2025). *Sem. + LLM Fea.* combines item semantic representations with these LLM-extracted item-text features. *Trace Emb. Only* uses only embeddings of the reasoning trace, without item semantic or feature-based inputs.

C.4 Matched Rollout-Count Control for Role-Based Aggregation

One potential concern is that the benefit of role-based aggregation may come simply from using more reasoning-model rollouts, rather than from the proficiency-conditioned solver profiles themselves. To separate the effect of additional rollouts from the effect of profile conditioning, we conduct a matched rollout-count control experiment on SAT Math.

Trace source	Setting	ACC	Macro-F1
QwQ	Single Direct	0.684	0.681
QwQ	Direct Replication	0.685	0.683
QwQ	Profiled Aggregation	**0.698**	**0.697**
Qwen	Single Direct	0.712	0.708
Qwen	Direct Replication	0.710	0.710
Qwen	Profiled Aggregation	**0.712**	**0.712**
Both	Single Direct	0.707	0.708
Both	Direct Replication	0.712	0.714
Both	Profiled Aggregation	**0.730**	**0.728**

Table 8: Matched rollout-count control on SAT Math. *Single Direct* uses one direct-prompt rollout, *Direct Replication* uses four independent generations from the same direct prompt, and *Profiled Aggregation* uses four proficiency-conditioned solver profiles. *Direct Replication* and *Profiled Aggregation* use the same number of reasoning-model rollouts, but only *Profiled Aggregation* introduces proficiency-conditioned profiles.

We compare three settings. *Single Direct* uses one direct-prompt reasoning trace from each reasoning model. *Direct Replication* uses four independent generations from the same direct prompt for each reasoning model. *Profiled Aggregation* uses four proficiency-conditioned solver profiles. Thus, *Direct Replication* and *Profiled Aggregation* use the same number of reasoning-model rollouts, but only *Profiled Aggregation* introduces proficiency-conditioned profiles. This matched comparison tests whether the observed benefit comes from additional stochastic samples alone or from structured profile-based aggregation.

Table 8 shows that simply increasing the number of stochastic direct-prompt rollouts does not explain the benefit of profile-based aggregation. For QwQ, moving from *Single Direct* to *Direct Replication* yields only a small gain, from 0.681 to 0.683 macro-F1, whereas *Profiled Aggregation* reaches 0.697. For Qwen, *Direct Replication* slightly improves macro-F1 over *Single Direct*, but *Profiled Aggregation* remains the best setting. For the combined trace-source setting, *Direct Replication* improves over *Single Direct*, but still remains below *Profiled Aggregation*, which achieves the best performance with 0.730 ACC and 0.728 macro-F1.

These results indicate that the gain from role-based aggregation is not merely an artifact of using more reasoning-model generations. Rather, proficiency-conditioned profiles introduce structured variation in the induced reasoning traces, leading to a more robust characterization of item difficulty than repeated direct-prompt sampling alone.

C.5 Representative Item-level Episode Profiles

ID	Diff.	Total	Think	Answer	% Impl.	% Anal.	% Ver.	% Read
# 217	Easy	557	399	158	0.253	0.151	0.172	0.120
# 235	Med	795	618	177	0.347	0.189	0.156	0.079
# 607	Hard	995	728	267	0.403	0.203	0.130	0.074

Table 9: **Characterization of representative SAT Math reasoning traces generated by Qwen3-32B.** As difficulty increases, total reasoning length and *Implement* share increase, while *Verify* and *Read* decrease in relative share; *Analyze* also increases.

Item	Prefix of collapsed episode trajectory (first 8 episodes)
Easy (# 217)	`Monitor(3)` → `Read(16)` → `Monitor(13)` → `Read(13)` → `Plan(34)` → `Implement(12)` → `Plan(4)` → `Verify(6)` . . .
Med (# 235)	`Read(29)` → `Monitor(2)` → `Verify(1)` → `Analyze(16)` → `Verify(10)` → `Plan(15)` → `Monitor(6)` → `Analyze(20)` . . .
Hard (# 607)	`Monitor(11)` → `Analyze(43)` → `Read(9)` → `Monitor(5)` → `Analyze(53)` → `Plan(5)` → `Implement(14)` → `Plan(5)` . . .

Table 10: **Prefixes of the collapsed episode sequences derived from Qwen3-32B reasoning traces for three representative SAT Math items.** For each item, we show the first 8 episodes in temporal order after collapsing consecutive identical labels and summing their token counts, split across two lines for readability. The Hard example already allocates substantial early budget to *Analyze*, whereas the Easy example remains more fragmented across monitoring, reading, planning, and short execution/checking steps.

Item

What is the value of $f(10)$?
The function f is defined by $f(x) = 4x - 3$.
Options:
(A) -30
(B) 37
(C) 40
(D) 43

Model Think

```
<think>\nOkay, let's see. I need to find the value of f(10) for the function f(x) = 4x - 3. Hmm,
↪  functions can sometimes be confusing, but I think this one is straightforward. So, the function is
↪  given as f(x) equals 4 times x minus 3. That means for any value of x, I just plug it into this
↪  equation. \n\nSo, if I want f(10), I should replace every x in the function with 10. Let me write
↪  that out. So f(10) would be 4 multiplied by 10, and then subtract 3. Let me calculate that. 4 times
↪  10 is 40, right? Then subtract 3 from 40. That would be 40 minus 3, which is 37. \n\nWait, let me
↪  check again. Maybe I made a mistake. Let me do the multiplication first. 4*10 is definitely 40.
↪  Then subtract 3. Yeah, 40 - 3 is 37. So the answer should be 37. Let me look at the options. Option
↪  B is 37. So that's probably the correct answer.\n\nBut let me make sure I didn't mix up anything.
↪  The function is f(x) = 4x - 3, not 4(x - 3). If it were 4(x - 3), then plugging in 10 would be
↪  4*(10 - 3) = 4*7 = 28. But that's not the case here. The original function is 4x minus 3, so order
↪  of operations is important here. There's no parentheses around the x - 3, so it's just 4 times x,
↪  minus 3. So yeah, 4*10 is 40, minus 3 is 37. \n\nI think that's right. Let me check once more.
↪  Maybe I can plug in another number to see if the function works as expected. For example, if x is 1,
↪  then f(1) should be 4*1 - 3 = 1. Let me see. 4 times 1 is 4, minus 3 is 1. That seems correct. If x
↪  is 2, 4*2 is 8, minus 3 is 5. So f(2) is 5. That makes sense. So applying the same logic, f(10) is
↪  4*10 - 3 = 40 - 3 = 37. Yep, that's consistent. \n\nTherefore, the correct answer should be option
↪  B, which is 37. I don't see any mistakes in my reasoning here. The key was substituting x with 10
↪  correctly and following the order of operations. Since there's no parentheses around the x in the
↪  original function, it's just multiplication and then subtraction. So all steps check out.\n</think>
```

Model Answer

```
To solve the problem, we are given a function and asked to evaluate it at a specific
↪  input.\n\n---\n\n### Step 1: Understand the Function\n\nThe function is defined as:\n\n$$\nf(x) =
↪  4x - 3\n$$\n\nThis is a linear function where the output is calculated by multiplying the input $
↪  x $ by 4 and then subtracting 3.\n\n---\n\n### Step 2: Substitute the Input\n\nWe are asked to
↪  find $ f(10) $. To do this, we substitute $ x = 10 $ into the function:\n\n$$\nf(10) = 4(10) -
↪  3\n$$\n\n---\n\n### Step 3: Perform the Arithmetic\n\n1. Multiply:  \n   $$\n   4 \\times 10 =
↪  40\n   $$\n\n2. Subtract:  \n   $$\n   40 - 3 = 37\n   $$\n\n---\n\n### Step 4: Match with the
↪  Correct Option\n\nThe result is 37. Looking at the answer choices:\n\n- (A) $-30$\n- (B) $37$\n-
↪  (C) $40$\n- (D) $43$\n\nThe correct answer is **(B) 37**.\n\n---\n\n### Final
↪  Answer:\n\n$$\n\\boxed{B}\n$$
```

Gold Answer: B **Difficulty:** Easy

Figure 2: SAT Math item # 217 (Easy), showing the full Qwen3-32B-generated thinking trace, final answer, gold answer, and difficulty label.

Item

What is the slope of line n?
When line n is graphed in the xy-plane, it has an x-intercept of $(-4, 0)$ and a y-intercept of $(0, \frac{86}{3})$.
Options:
(A) $\frac{3}{344}$
(B) $\frac{6}{\frac{40}{3}}$
(C) $\frac{43}{6}$
(D) $\frac{344}{3}$

Model Think

```
<think>\nOkay, so I need to find the slope of line n. The problem says that the line has an x-intercept at (-4, 0)
↪  and a y-intercept at (0, 86/3). Hmm, slope... right. The slope of a line can be found if you have two points on
↪  the line. Since they gave me both intercepts, those are two points. So I can use the formula for slope, which is
↪  (y2 - y1)/(x2 - x1). Let me recall that formula correctly. Yeah, slope is rise over run, so the difference in y
↪  divided by the difference in x between two points.\n\nAlright, let me label the points. Let's take the
↪  x-intercept as (-4, 0). Let me call that (x1, y1). And the y-intercept is (0, 86/3), so that would be (x2, y2).
↪  Wait, actually, does it matter which one I choose as first or second? I think not, as long as I'm consistent. But
↪  let me stick with this.\n\nSo plugging into the formula: slope m = (y2 - y1)/(x2 - x1). So substituting the
↪  values, y2 is 86/3, y1 is 0. So numerator is 86/3 - 0 = 86/3. Denominator is x2 - x1, which is 0 - (-4) = 0 + 4 =
↪  4. So the slope is (86/3)/4. Dividing by 4 is the same as multiplying by 1/4, so 86/3 * 1/4 = 86/(12). Let me
↪  simplify that. 86 divided by 12... Let's see if they have a common factor. Both are even numbers, so divide
↪  numerator and denominator by 2: 43/6. So the slope is 43/6. Let me check the answer choices. Option C is 43/6. So
↪  that must be the answer. \n\nWait, let me double-check my steps. First, the two points are correct. The
↪  x-intercept is where y=0, so (-4,0), and the y-intercept is where x=0, so (0, 86/3). Correct. Then slope formula:
↪  change in y over change in x. From (-4,0) to (0, 86/3), the change in y is 86/3 - 0 = 86/3. The change in x is 0 -
↪  (-4) = 4. So slope is (86/3)/4. Yes, that's 86/12, simplifies to 43/6. Yep. So option C is correct. Let me check
↪  if any other options could be equivalent. Option B is 6 divided by (40/3)... Wait, that would be 6*(3/40)= 18/40
↪  = 9/20, which is not 43/6. Option A is 3/344, which is way smaller. Option D is 344/3, which is much larger. So
↪  only option C matches. So I think that's correct. \n\nAlternatively, maybe I can think of the equation of the
↪  line. Since I have intercepts, the intercept form of a line is x/a + y/b = 1, where a is the x-intercept and b is
↪  the y-intercept. So here, a is -4? Wait, intercept form is usually for positive intercepts. Wait, actually, the
↪  intercept form is x divided by the x-intercept plus y divided by the y-intercept equals 1. But if the x-intercept
↪  is negative, does that affect it? Let me think. If the x-intercept is (-4, 0), then a is -4. So plugging into
↪  intercept form: x/(-4) + y/(86/3) = 1. Maybe that's more complicated. Alternatively, converting to
↪  slope-intercept form. Since the y-intercept is given as (0, 86/3), so in y = mx + b, b is 86/3. So if I can find
↪  m. Well, we already did that with the slope formula. But maybe using another method to confirm. Let's take the
↪  point (-4, 0) and plug into y = mx + b. So 0 = m*(-4) + 86/3. Solving for m: -4m = -86/3 => m = (-86/3)/(-4) =
↪  (86/3)/4 = same as before, 86/12 = 43/6. Yep, same answer. So definitely, the slope is 43/6, which is option C.
↪  \n\nTherefore, the correct answer is C.\n</think>
```

Model Answer

```
To determine the slope of line $ n $, we are given two key points on the line:\n\n- The **x-intercept** is $ (-4, 0)
↪  $\n- The **y-intercept** is $ \\left(0, \\frac{86}{3}\\right) $\n\n---\n\n### Step 1: Use the slope
↪  formula\n\nThe slope $ m $ of a line passing through two points $ (x_1, y_1) $ and $ (x_2, y_2) $ is given
↪  by:\n\n$$\nm = \\frac{y_2 - y_1}{x_2 - x_1}\n$$\n\nLet's label the points:\n\n- $ (x_1, y_1) = (-4, 0) $\n- $
↪  (x_2, y_2) = \\left(0, \\frac{86}{3}\\right) $\n\nSubstitute into the formula:\n\n$$\nm = \\frac{\\frac{86}{3} -
↪  0}{0 - (-4)} = \\frac{\\frac{86}{3}}{4}\n$$\n\n---\n\n### Step 2: Simplify the
↪  expression\n\n$$\n\\frac{\\frac{86}{3}}{4} = \\frac{86}{3} \\cdot \\frac{1}{4} = \\frac{86}{12}\n$$\n\nNow
↪  simplify $ \\frac{86}{12} $ by dividing both numerator and denominator by their greatest common divisor, which is
↪  2:\n\n$$\n\\frac{86 \\div 2}{12 \\div 2} = \\frac{43}{6}\n$$\n\n---\n\n### Step 3: Match with the answer
↪  choices\n\nThe simplified slope is $ \\frac{43}{6} $, which matches **Option C**.\n\n---\n\n### Final
↪  Answer:\n\n$$\n\\boxed{\\text{C}}\n$$
```

Gold Answer: C **Difficulty:** Medium

Figure 3: SAT Math item # 235 (Medium), showing the full Qwen3-32B-generated thinking trace, final answer, gold answer, and difficulty label.

Item

What is the value of a?
Function f is defined by $f(x) = -a^x + b$, where a and b are constants. In the xy-plane, the graph of $y = f(x) - 12$ has a y-intercept at $(0, -\frac{75}{7})$. The product of a and b is $\frac{320}{7}$.

Model Think

<think>\nOkay, let's try to figure out this problem step by step. So, the question is asking for the value of 'a' given some information about the function f(x) = -aˆx + b. There are two pieces of information provided: first, the graph of y = f(x) - 12 has a y-intercept at (0, -75/7). Second, the product of a and b is 320/7. Alright, let's break this down.\n\nFirst, let me recall that the y-intercept of a graph occurs where x = 0. So, if the graph of y = f(x) - 12 has a y-intercept at (0, -75/7), that means when we plug in x = 0 into the function y = f(x) - 12, the result should be -75/7. Let me write that down.\n\nSo, y = f(0) - 12 = -75/7. Therefore, f(0) - 12 = -75/7. Let me solve for f(0). If I add 12 to both sides, f(0) = -75/7 + 12. Hmm, I need to compute that. Let me convert 12 to sevenths. 12 is 84/7, right? So, -75/7 + 84/7 is (84 - 75)/7 = 9/7. Therefore, f(0) = 9/7.\n\nNow, since f(x) is defined as -aˆx + b, let me compute f(0). For any exponential function aˆx, when x = 0, aˆ0 = 1, right? So, f(0) = -1 + b. Therefore, -1 + b = 9/7. Let me solve for b. Adding 1 to both sides gives b = 9/7 + 1. Again, converting 1 to sevenths: 7/7. So, 9/7 + 7/7 = 16/7. Therefore, b = 16/7.\n\nOkay, so now we have b. The problem also states that the product of a and b is 320/7. So, a * b = 320/7. Since we found that b is 16/7, we can substitute that into the equation. Let me write that:\n\na * (16/7) = 320/7.\n\nTo solve for a, divide both sides by (16/7). Dividing by a fraction is the same as multiplying by its reciprocal, so:\n\na = (320/7) / (16/7) = (320/7) * (7/16). The 7s cancel out, so we have 320/16. Let me compute that. 320 divided by 16. Well, 16 times 20 is 320, so 320/16 is 20. Therefore, a = 20. \n\nWait, that seems straightforward, but let me check my steps again to make sure I didn't make a mistake. Let's verify each step.\n\nFirst, starting with the y-intercept. The graph is y = f(x) - 12. At x = 0, y is -75/7. Therefore, f(0) - 12 = -75/7. Solving for f(0): f(0) = -75/7 + 12. Converting 12 to 84/7, so 84/7 - 75/7 = 9/7. That seems right. Then f(0) = -aˆ0 + b = -1 + b. So, -1 + b = 9/7. Therefore, b = 9/7 + 1. Which is 16/7. Correct. Then, a * b = 320/7. Substituting b = 16/7, so a = (320/7) / (16/7) = 20. That seems correct. \n\nBut let me check if there's any other condition or something I might have missed. The function is f(x) = -aˆx + b. The problem didn't mention anything else about the graph, like another point or behavior, but we were given the product of a and b. So with the given information, we were able to find b first, then use the product to find a. \n\nIs there a possibility that a could be negative? Well, in exponential functions, the base a is typically positive and not equal to 1. Since it's an SAT problem, they probably expect a positive real number for a. Also, if a were negative, then aˆx might not be a real number for all x, unless x is an integer. But since the problem doesn't specify any restrictions, and given that the answer comes out nicely as 20, which is positive, I think that's the correct answer.\n\nLet me just verify once again with the numbers. If a is 20 and b is 16/7, then their product is 20*(16/7) = 320/7, which matches the given condition. Then f(0) is -1 + 16/7 = 9/7, so y = f(x) -12 at x=0 is 9/7 - 12. Let me compute that again. 9/7 - 12 is 9/7 - 84/7 = -75/7, which is exactly the y-intercept given. So all the numbers check out. \n\nTherefore, I think the value of a is 20. \n\n**Final Answer**\nThe value of $ a $ is \\boxed{20}.\n</think>

Model Answer

To determine the value of $ a $, we are given the function:\n\n$$\nf(x) = -aˆx + b\n$$\n\nand two key pieces of information:\n\n1. The graph of $ y = f(x) - 12 $ has a y-intercept at $ (0, -\\frac{75}{7}) $.\n2. The product $ ab = \\frac{320}{7} $.\n\n---\n\n### Step 1: Use the y-intercept to find $ f(0) $\n\nThe y-intercept occurs when $ x = 0 $. So, we substitute $ x = 0 $ into the expression $ y = f(x) - 12 $:\n\n$$\ny = f(0) - 12 = -\\frac{75}{7}\n$$\n\nSolving for $ f(0) $:\n\n$$\nf(0) = -\\frac{75}{7} + 12\n$$\n\nConvert 12 to a fraction with denominator 7:\n\n$$\n12 = \\frac{84}{7}\n$$\n\nSo:\n\n$$\nf(0) = \\frac{84 - 75}{7} = \\frac{9}{7}\n$$\n\n---\n\n### Step 2: Use the definition of $ f(x) $ to find $ b $\n\nRecall that:\n\n$$\nf(x) = -aˆx + b\n$$\n\nSo:\n\n$$\nf(0) = -aˆ0 + b = -1 + b\n$$\n\nSet this equal to the value we just found:\n\n$$\n-1 + b = \\frac{9}{7}\n$$\n\nSolving for $ b $:\n\n$$\nb = \\frac{9}{7} + 1 = \\frac{9}{7} + \\frac{7}{7} = \\frac{16}{7}\n$$\n\n---\n\n### Step 3: Use the product $ ab = \\frac{320}{7} $ to find $ a $\n\nNow that we know $ b = \\frac{16}{7} $, substitute into the product:\n\n$$\na \\cdot \\frac{16}{7} = \\frac{320}{7}\n$$\n\nSolve for $ a $:\n\n$$\na = \\frac{320}{7} \\div \\frac{16}{7} = \\frac{320}{7} \\cdot \\frac{7}{16} = \\frac{320}{16} = 20\n$$\n\n---\n\n### Final Answer\n\n$$\n\\boxed{20}\n$$

Gold Answer: 20 **Difficulty:** Hard

Figure 4: SAT Math item # 607 (Hard), showing the full Qwen3-32B-generated thinking trace, final answer, gold answer, and difficulty label.

D Additional Analysis Visualizations

In this appendix, we provide the full role-wise visualizations of the episode-induced process representation on SAT Math using Qwen3-32B generated reasoning traces. Each figure corresponds to one solver role (*direct*, *weak*, *medium*, or *strong*) and combines three complementary views: (1) the length summary across Easy, Medium, and Hard items, (2) the episode-ratio deltas between difficulty groups, and (3) the transition-delta heatmaps between difficulty groups.

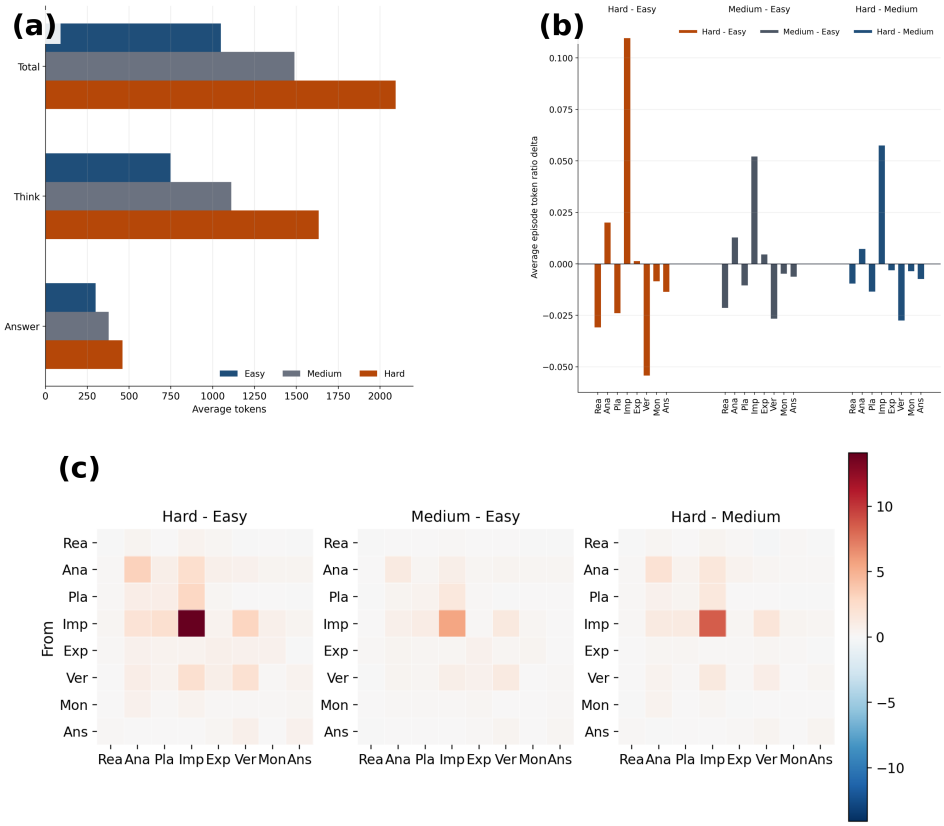


Figure 5: Role-wise visualization for the *direct* solver profile on SAT Math. The figure combines the length summary, episode-ratio deltas, and transition-delta heatmaps.

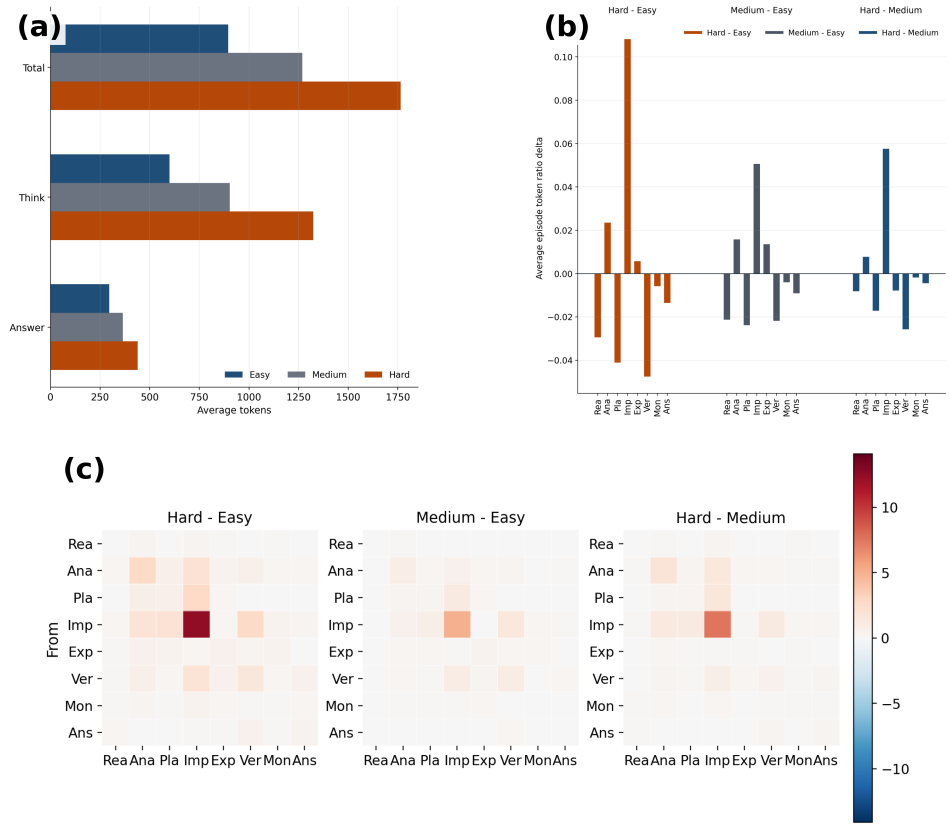


Figure 6: Role-wise visualization for the *weak* solver profile on SAT Math. The figure combines the length summary, episode-ratio deltas, and transition-delta heatmaps.

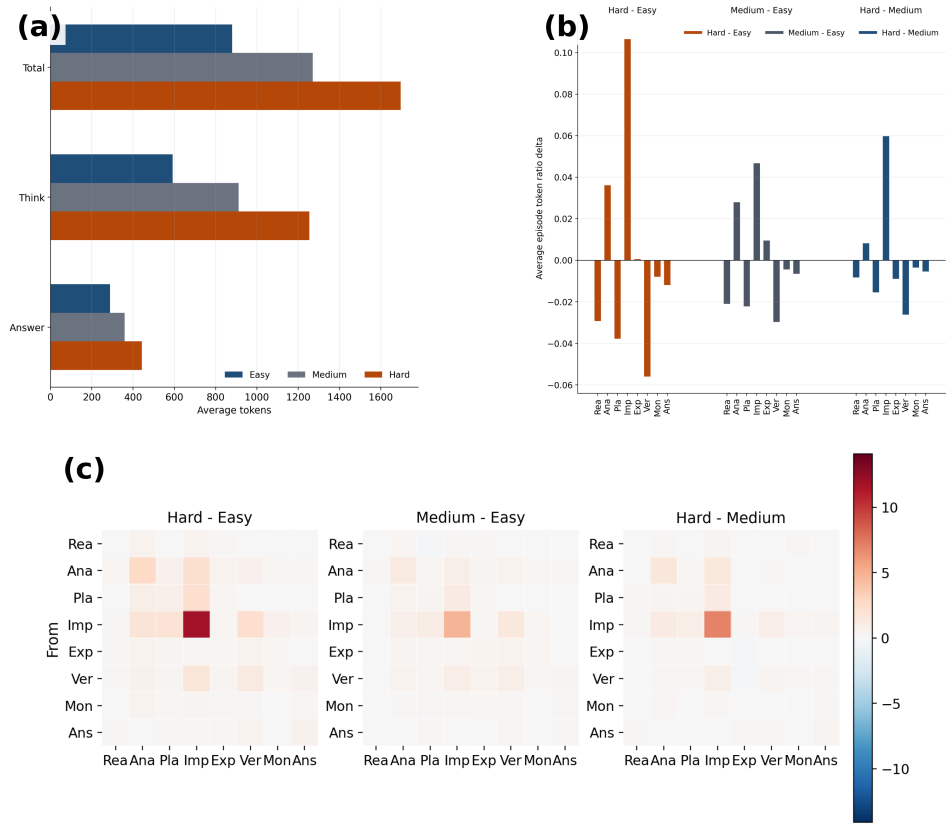


Figure 7: Role-wise visualization for the *medium* solver profile on SAT Math. The figure combines the length summary, episode-ratio deltas, and transition-delta heatmaps.

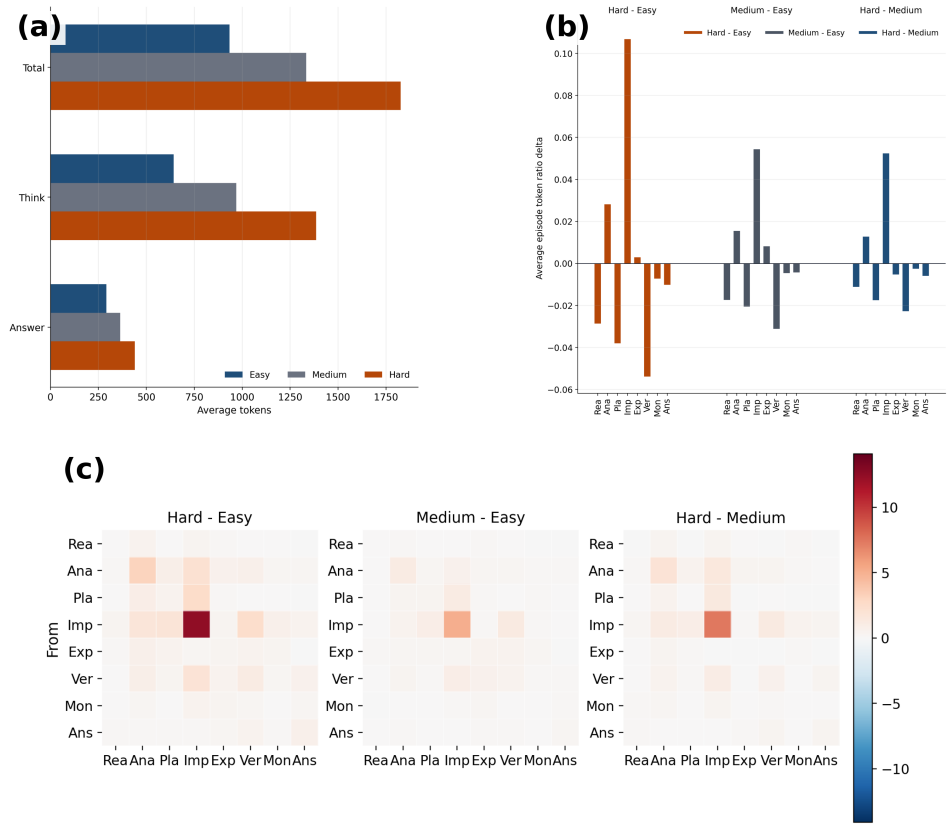


Figure 8: Role-wise visualization for the *strong* solver profile on SAT Math. The figure combines the length summary, episode-ratio deltas, and transition-delta heatmaps.