

Design Choices in Encoder-Based IRT-3PL Parameter Prediction for Passage-Based Multiple-Choice Questions

Sungjin Nam

ACT Education Corp.

500 Act Dr

Iowa City, IA 52243

sungjin.nam@act.org

Abstract

We examine model design choices for encoder-based prediction of IRT-3PL item parameters in reading comprehension tasks with passage-based multiple-choice questions (MCQs). We compare three aspects: (1) encoding strategies for passage and question texts, including optional cross-attention, (2) per-parameter vs. joint-parameter prediction with MSE and CRPS losses, and (3) random vs. passage-grouped batching. Results show that separate-then-fuse encoding is substantially more efficient than concatenated-input encoding while maintaining comparable parameter recovery. Cross-attention improves recovery for discrimination and guessing but weakens difficulty recovery. Joint-parameter prediction achieves performance close to per-parameter prediction in the strongest configurations while reducing training time. CRPS is most useful for joint prediction, and passage-grouped batching consistently improves recovery of the guessing parameter. Overall, joint prediction with CRPS and passage-grouped batching provides the most balanced performance while maintaining the efficiency of separate-then-fuse encoding without cross-attention, showing that the proposed model design can support efficient IRT parameter prediction for passage-based MCQs.

1 Introduction

Collecting thousands of student responses for field testing is common practice in educational assessment for estimating item parameters, such as difficulty and discrimination, before operational use. These responses are needed to cover the item pool and ensure stable parameter estimates, but collecting them can be costly and time-consuming. Multiple studies have shown that encoder-based linguistic representations of item texts can be used to predict IRT item parameters (Kapoor et al., 2025) and reduce the need for extensive field testing (Yancey et al., 2024). However, studies that predict item

parameters for passage-based multiple-choice questions (MCQs) remain limited, either by restricting each question to a single short passage (Kapoor et al., 2025) or by exploring model architecture only as a simple per-parameter regressor (Yancey et al., 2024).

In this study, we investigate design choices for encoder-based item parameter prediction models, with a focus on passage-based MCQs in reading comprehension tasks. We focus on three main design questions: (1) the encoding strategy for passage and question texts, (2) the choice between per- and joint-parameter prediction with different loss functions, and (3) the design of training batches. We examine these questions to inform design choices that can improve the direct prediction of IRT-3PL parameters, namely item discrimination (*a*), difficulty (*b*), and guessing (*c*).

Passage-based MCQs introduce several challenges that are less prominent in standalone MCQ items. First, passage-based MCQs require the model to capture interactions between a passage and multiple associated questions. Second, when multiple questions share the same reading passage, the model needs to capture both common passage-level features and item-specific variations. Third, long passages increase pressure on computational budgets. To address these issues, we explore model design choices for predicting item parameters while maintaining computational efficiency.

More specifically, we address the following research questions:

- RQ1: How does the encoding strategy affect the prediction quality of item parameters, measured by Pearson correlation and root mean squared error (RMSE), and model training efficiency?
- RQ2: How does adding cross-attention between passage and question representations affect item parameter prediction quality?

- RQ3: How does the joint-parameter prediction design compare to per-parameter prediction in terms of prediction quality and efficiency?
- RQ4: How does the choice of loss function affect the prediction quality of item parameters?
- RQ5: How does the design of training batches influence the prediction of item parameters?

The contributions of this paper are threefold. First, we show that passage-based MCQ item parameter prediction can be improved by reflecting the *unique challenges of the dataset* in the model design. RQ1 examines whether encoding passages and questions separately (*separate-then-fuse encoding*) is a viable alternative to encoding them together (*concatenated-input encoding*), especially under long-passage conditions (~ 760 words long). RQ2 further examines how an optional cross-attention layer, which allows the item representation to attend to passage tokens before pooling, affects prediction quality. RQ5 explores whether placing items linked to the same passage in a *passage-grouped* batch improves prediction performance without additional training cost by leveraging the within-passage structure of items.

Second, we show that *both efficiency and effectiveness* in item parameter prediction can be achieved with carefully designed model architectures and training strategies. When training an encoder-based model, input token length and the number of item parameters to predict are two main factors that can affect training time. RQ1 investigates whether prediction quality can be preserved while reducing the maximum input-token length, which leads to a non-linear reduction in model training and inference time. RQ3 examines whether the joint-parameter prediction design, which predicts all three IRT parameters simultaneously, is more efficient than per-parameter prediction, which requires multiple model runs to collect predictions for each parameter, while preserving prediction quality. RQ4 compares mean squared error (MSE), a point prediction loss, with the continuous ranked probability score (CRPS), a distributional loss, to see if the latter can improve prediction quality by encouraging the model to capture uncertainty in its predictions.

Third, compared to existing studies that focused on predicting item parameters for simpler IRT mod-

els, such as 1PL or 2PL models (Maeda, 2025; Ormerod, 2026), we extensively compare model design choices for predicting all three IRT-3PL parameters. This setting is more challenging than 1PL or 2PL prediction because it includes the guessing parameter, and our results demonstrate the feasibility of predicting all three item parameters accurately.

2 Related Work

2.1 Predicting Item Parameters with ML Models

Prior work on educational response modeling includes both traditional psychometric simulation and recent machine learning approaches. In IRT-based simulation, researchers specify or estimate item parameters and student ability distributions, then generate response probabilities from the assumed model. This line of work remains a common baseline because it is well established in educational measurement, but it depends on having plausible parameter values in advance.

Recent NLP-based work has attempted to model student responses from item texts. Lalor et al. (2019) used LSTM-based supervised models to simulate lower- and higher-skilled students. Other studies have used groups of fine-tuned language models at various training stages (Maeda, 2025) or off-the-shelf models with different capabilities (Calais et al., 2024) to simulate authentic student responses at varying skill levels. Hybrid approaches, such as reconstructing item characteristic curves from simulated responses (Ormerod, 2026), have shown potential to recover item parameters with fine-tuned decoder-based models. Prompt-engineering approaches have used LLMs to simulate student profiles, such as knowledge components (Lu and Wang, 2024) or skill levels (Benedetto et al., 2024), to generate responses and analyze item properties. While simulation-based approaches are useful for capturing variation in student responses across skill levels (Maeda, 2025; Lalor et al., 2019), the alignment between simulated responses and actual human responses or psychometric properties has been limited.

A more straightforward approach is to estimate item parameters directly from item texts. Yancey et al. (2024) used BERT-based models to estimate IRT parameters for items and showed that model outputs can substantially reduce field-testing requirements. Kapoor et al. (2025) examined the use

of item text embeddings and manual annotations to predict IRT parameters. However, these studies focused on standalone items with short passages, and they did not explore design choices for passage-based MCQs. Our work builds on this direct-prediction setting. Passage-based MCQs create a more structured prediction problem than standalone items because the model must represent both the passage context and the local question-answer interaction. We specifically investigate combinations of encoding and training strategies for predicting item parameters in substantially longer passage-based MCQ items.

2.2 Joint Prediction in Assessment-Related Tasks

Joint prediction, or multi-task learning, where a single model predicts multiple outputs, is a popular approach for improving predictive model performance by sharing representations across related outcomes while also improving model efficiency. Studies have explored applications of joint prediction in various educational tasks. For example, studies on automated essay scoring explored the benefits of joint prediction of multiple outcomes, such as essay scores and off-topic labels, to improve scoring accuracy and produce essay-response representations that are aware of different writing prompts (Nam, 2025). Studies on item parameter prediction have also explored joint prediction of IRT-2PL parameters, namely discrimination and difficulty, from text and image inputs (Maeda and Lu, 2026). In this study, we investigate the benefits of joint prediction for *IRT-3PL* item parameters, which have not yet been explored extensively. We examine design choices for jointly predicting multiple item parameters, including prediction layer design and loss function comparisons, to improve the efficiency and effectiveness of item parameter prediction.

3 Dataset

We used data from a reading comprehension assessment administered to high school students. A defining characteristic of the dataset is its passage-centered structure. First, the dataset organizes questions into passage-linked sets. Each passage is associated with 10 MCQs, and four passages are grouped into a form, which is a single test unit administered to each student. Second, the passages are relatively long (766.51 words on average), com-

pared to other MCQ benchmarks. Thus, to accurately estimate item parameter values, the model needs to capture both global characteristics of a long passage and finer-grained variation among questions attached to the same passage.

The entire dataset contains 43 forms (172 passages and 1,720 items). All reported results use the same predefined five-fold cross-validation setup as the experimental code. In each fold, models are trained and evaluated on form-level subsets. Four folds contained 31, 3, and 9 forms in the training, development, and test sets, respectively, and one fold contained 33, 3, and 7 forms. The prediction targets are the calibrated IRT-3PL parameters for discrimination (a), difficulty (b), and guessing (c). All calibrated parameters were equated, so we could place items from different forms on a common scale and make the cross-fold comparisons psychometrically meaningful.

4 Model Design

We treated the item parameter prediction task as a regression problem, in which the model takes the passage and question texts as input and outputs predicted IRT-3PL parameters. All experiments used the ModernBERT-base (Warner et al., 2025) encoder as the backbone. We fine-tuned the last six encoder layers and kept the remaining layer weights frozen. Encoder outputs were projected to a 384-dimensional hidden space before being passed to the regression layer. The regression head consisted of Linear-GELU-Dropout (0.1)-Linear layers. Across conditions, training used AdamW with a head learning rate of 3×10^{-4} , an encoder learning rate of 1×10^{-5} , cosine learning-rate decay, weight decay of 0.03, a warmup ratio of 0.05, seven epochs, and bfloat16 training.

Each item was formatted as the question stem followed by labeled answer options. Passages and items were tokenized either separately or as one concatenated sequence. The token budgets were 1512 tokens for passages, 512 for items, and 2024 for the concatenated-input condition. The model added a learned embedding of the correct answer key (A–D) before pooling, so the final representation could use both the text content and the answer-key position. Figure 1 provides additional details on the model designs.

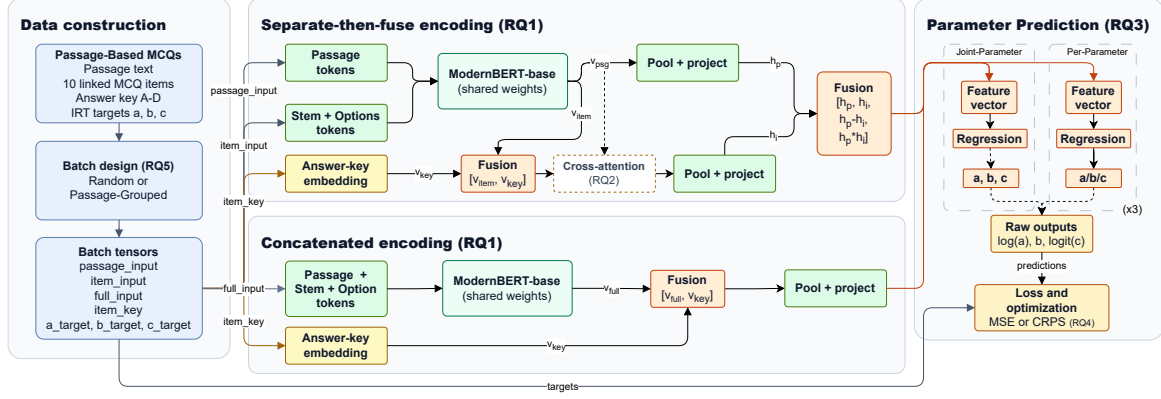


Figure 1: Model design for passage-based MCQ item parameter prediction. We tested IRT-3PL parameter prediction when the passage and question were encoded together or separately (RQ1, RQ2), when the model predicted parameters jointly or separately (RQ3), and when the model was trained with different loss functions (RQ4). We also compared random batching and passage-grouped batching (RQ5).

4.1 Encoding Strategies

The first model design component we compared was encoding strategies for the passage and question texts. For the encoding comparison, we implemented two alternatives (RQ1). In the *concatenated-input* encoding condition, the passage and item texts were concatenated and passed through ModernBERT in one forward pass, after which the pooled sequence representation was projected and sent to the regression head. In the *separate-then-fuse* encoding condition, the same shared encoder processed the passage and the item in two passes. The resulting pooled passage and item vectors, h_p and h_i , were projected separately and fused as a concatenation of $[h_p, h_i, h_p - h_i, h_p * h_i]$. We also tested a cross-attention layer (RQ2), in which item tokens attend to passage tokens from separate passes through a two-layer TransformerDecoder before pooling to generate a passage-attentive representation for h_i . All encoded representations used mean pooling, based on preliminary analyses.

4.2 Joint Prediction and Loss Function

Second, we compared the joint-parameter prediction design (RQ3). In the *per-parameter* condition, three independent models were trained, and each model with an encoder backbone and regression head predicted only one of a , b , or c . In the *joint-parameter* condition, one model predicted all three parameters simultaneously. The output parameterization follows the IRT constraints used in the model code. The regression head predicted $\log a$, b , and $\text{logit}(c)$ on unconstrained scales, then trans-

formed them back to the IRT space with $\exp(a)$ and $\sigma(c)$.

We also tested multiple loss functions for model training (RQ4). Let $\mathcal{P} = \{a, b, c\}$ denote the target item parameters, and let $y_{i,p}$ and $\hat{y}_{i,p}$ be the observed and predicted values for item i and item parameter p . The standard MSE objective was implemented as

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^N \sum_{p \in \mathcal{P}} (y_{i,p} - \hat{y}_{i,p})^2 \quad (1)$$

In addition to MSE, we implemented CRPS, a distributional loss that encourages the model to capture uncertainty in its predictions. For CRPS, each parameter was modeled as a Gaussian distribution on its raw prediction scale. With raw-scale target $\tilde{y}_{i,p}$ (e.g., $\log a$ or $\text{logit}(c)$), predicted mean $\mu_{i,p}$, and standard deviation $\sigma_{i,p}$, we calculated $z_{i,p} = (\tilde{y}_{i,p} - \mu_{i,p}) / \sigma_{i,p}$. The objective was implemented as

$$\mathcal{L}_{\text{CRPS}} = \frac{1}{N} \sum_{i=1}^N \sum_{p \in \mathcal{P}} \sigma_{i,p} \left[z_{i,p} (2\Phi(z_{i,p}) - 1) + 2\phi(z_{i,p}) - \frac{1}{\sqrt{\pi}} \right] \quad (2)$$

where ϕ and Φ are the standard normal density and cumulative distribution functions. Under CRPS, each output dimension was doubled so that the head predicted both a mean and a log-variance, and the loss was computed on the raw Gaussian scale: $\log a$ for discrimination, b for difficulty, and $\text{logit}(c)$ for guessing.

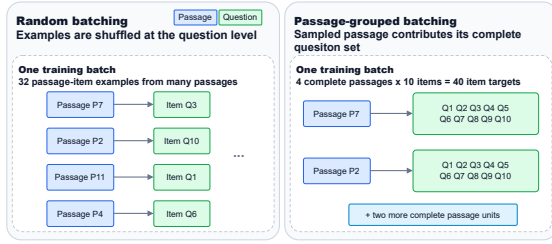


Figure 2: Comparison of random batching and passage-grouped batching (RQ5). Random batching samples shuffled individual passage-item examples from many passages. Within-passage item structure is usually split across updates. Passage-grouped batching samples complete passage-item units. All questions linked to a passage are used in the same optimizer update step. The passage representation can be reused within the batch.

4.3 Batch Design

The third component was a comparison between passage-grouped batching and random batching (RQ5). The *passage-grouped* condition constructed each microbatch from a complete passage-item unit, in which one passage was linked with 10 MCQ items, so all questions belonging to a sampled passage were optimized together. For training, we used four passages (40 questions) per batch. In the *random* batching condition, individual questions were shuffled and trained with a batch size of 32. The passage-grouped condition was designed to provide more efficient training by reusing the passage encoding across questions from the same passage. It also allowed the model to learn within-passage structure during a training step. In comparison, the random batch condition required separate passage encodings for each question, with less chance of seeing the questions from the same passage together in a training batch (Figure 2).

5 Results

5.1 Encoding Strategies

Following previous studies (Kapoor et al., 2025; Ormerod, 2026), we evaluate prediction recovery quality using Pearson correlation and RMSE. To answer RQ1, we compared *concatenated-input* and *separate-then-fuse* encoding strategies. For RQ2, we examined the effect of adding a cross-attention (CA) module for the *separate-then-fuse* condition that lets item tokens attend to passage tokens before pooling. We kept per-parameter prediction and random batching conditions fixed. Table 1 reports the MSE encoding comparison. CRPS results are included in Appendix B.

The results identified *separate-then-fuse* encoding as an efficient backbone with comparable performance, while CA mainly changed the balance among the three parameter-specific correlations. For RQ1, *separate-then-fuse* encoding (CA:x) provided comparable performance to *concatenated-input* encoding (e.g., 0.383 vs. 0.389 average Pearson). *Separate-then-fuse* improved *b* recovery, while *a* and *c* were slightly better with *concatenated-input* encoding. For RQ2, adding cross-attention to *separate-then-fuse* produced the best overall performance, improving *a* (e.g., Pearson *r* of 0.514 $\rightarrow$ 0.533) and *c* (0.189 $\rightarrow$ 0.333), with a small drop in *b* (0.445 $\rightarrow$ 0.433).

Efficiency differences were large. *Concatenated-input* took $\sim 6\times$ longer to train than *separate-then-fuse* (5,047 vs. 804 seconds). Adding CA raised *separate-then-fuse* training time by $\sim 1.4\times$ (1,101 seconds).

5.2 Joint Prediction, Loss Functions, and Batch Design

Table 2 compares per- vs. joint-parameter prediction, random vs. passage-grouped batching, and MSE vs. CRPS loss under *separate-then-fuse* encoding without cross-attention. We focus on the non-CA setting in the main text for brevity. Appendices A and B summarize the corresponding CA results and the results for all configurations.

Overall, the combination of joint-parameter prediction, CRPS loss, and passage-grouped batching produced the best balanced Pearson performance in the non-CA comparison. For RQ3, *joint-parameter* prediction achieved performance comparable to *per-parameter* prediction in the strongest configurations. Under passage-grouped batching with CRPS loss and no cross-attention, the joint model was close to the per-parameter setting on *a* (e.g., 0.524 vs. 0.521) and *c* (0.338 vs. 0.332) while trailing slightly on *b* (0.456 vs. 0.462). It also reduced training and inference time by $\sim 3\times$ because a single model could predict all parameters simultaneously (814 vs. 270 seconds).

CRPS (RQ4) was most helpful for the joint-parameter model. Under random batching, switching from MSE to CRPS substantially improved the joint model’s *c* recovery (e.g., 0.110 $\rightarrow$ 0.226). It also improved *a* recovery for the joint model (0.450 $\rightarrow$ 0.529), while its effect on *b* was smaller and negative (0.442 $\rightarrow$ 0.427). Under passage-grouped batching, CRPS also improved joint-model recovery for *a*, *b*, and *c* (*a*: 0.517 $\rightarrow$ 0.524;

Encoding	CA	Pearson r ($\uparrow$)				RMSE ($\downarrow$)				Time ($\downarrow$)
		a	b	c	Avg.	a	b	c	Avg.	
Concat.	x	<u>0.532</u>	0.432	<u>0.204</u>	<u>0.389</u>	<u>0.291</u>	<u>0.854</u>	0.075	<u>0.407</u>	x6.3
Sep-Fuse.	x	0.514	0.445	0.189	0.383	0.302	0.849	0.075	0.409	x1.0
	o	0.533	<u>0.433</u>	0.333	0.433	0.282	0.870	0.075	0.409	<u>x1.4</u>

Table 1: Comparison of concatenated-input (Concat.) vs. separate-then-fuse (Sep-Fuse.) encoding. All settings used per-parameter prediction with random batching and MSE loss. CA indicates whether cross-attention was used. Separate-then-fuse encoding substantially reduced training time while maintaining comparable performance to concatenated-input encoding. Cross-attention improved a and c recovery, but reduced b recovery. Each metric’s **best** and second-best values are marked.

Batching	Loss	Prediction	Pearson r ($\uparrow$)				RMSE ($\downarrow$)			
			a	b	c	Avg.	a	b	c	Avg.
Random	MSE	Per-Param	0.514	0.445	0.189	0.383	0.302	0.849	0.075	0.409
		Joint-Param	0.450	0.442	0.110	0.334	0.290	0.855	0.075	0.407
	CRPS	Per-Param	0.522	0.444	0.249	0.405	0.290	0.846	0.079	0.405
		Joint-Param	0.529	0.427	0.226	0.394	0.289	0.895	0.079	0.421
Passage-Grouped	MSE	Per-Param	0.510	0.452	0.340	0.434	0.269	0.843	0.070	0.394
		Joint-Param	0.517	0.444	0.269	0.410	<u>0.270</u>	0.848	0.072	0.397
	CRPS	Per-Param	0.521	0.462	0.332	<u>0.438</u>	0.271	0.843	<u>0.071</u>	<u>0.395</u>
		Joint-Param	<u>0.524</u>	<u>0.456</u>	<u>0.338</u>	0.439	<u>0.270</u>	<u>0.844</u>	<u>0.071</u>	<u>0.395</u>

Table 2: Results for combinations of batching, loss function, and prediction settings under separate-then-fuse encoding without cross-attention. Joint-parameter prediction with CRPS loss and passage-grouped batching produced the best overall performance, while also reducing training time by $\sim 3\times$ compared to per-parameter prediction. Each column’s **best** and second-best values across all rows are marked.

b : 0.444 $\rightarrow$ 0.456; c : 0.269 $\rightarrow$ 0.338).

For RQ5, passage-grouped batching substantially improved c recovery in every condition. Without cross-attention, it also consistently improved b , while the effect on a was mixed. For example, for the joint-parameter, CRPS, non-CA configuration, grouped batching raised b from 0.427 to 0.456 and c from 0.226 to 0.338 while leaving a nearly unchanged (0.529 vs. 0.524). It also provided slightly greater improvements for the joint-parameter model than for the per-parameter model.

6 Discussion and Conclusion

The findings suggest that item-parameter prediction for passage-based MCQs benefits from aligning model design with the structure of the task. The strongest settings did not rely on a single change. They combined a computationally efficient passage-item encoder, a joint prediction head, a distributional loss, and batches that preserved passage-level item grouping.

The efficiency result for *separate-then-fuse* encoding is important when passage-based items contain long shared contexts. Concatenating the pas-

sage and item gives the encoder direct access to all tokens at once, but the longer sequence substantially increases cost. In our experiments, it did not yield a clear parameter-recovery advantage. The separate-then-fuse design therefore appears to retain enough passage-item information while avoiding repeated long-sequence encoding in encoder-based item parameter models. However, this conclusion may not transfer to larger or decoder-based models, where longer concatenated inputs can be better represented and provide richer token-level interactions.

The cross-attention results show that adding an explicit interaction module is not automatically beneficial. CA improved recovery for a and c , but it weakened b and introduced instability for c under MSE with joint prediction. The extra passage-item interaction capacity may make optimization more sensitive to selected item-level signals. In our setting, CA is better treated as an optional component when recovering a or c is more important than preserving b performance and training simplicity.

The joint-prediction, CRPS, and passage-grouped batching results point to a complementary set of design choices. Joint prediction reduces

training cost by replacing three separate models with one shared model, but it was most vulnerable on c , the parameter with the most fragile recovery in these experiments. CRPS helped this setting by modeling each target as a distribution rather than a point estimate. Passage-grouped batching also improved the results from these settings by presenting questions from the same passage in the same optimization step. These results suggest that uncertainty-aware objectives and passage-aware batching could make efficient joint prediction viable for passage-based item parameter recovery.

Some items could have more extreme parameter values than others, and using these items in the training set might introduce noise into the training process. Although this study did not conduct a detailed analysis of item-level characteristics, filtering out items with extreme parameters might improve the quality of training items and help the model learn meaningful patterns from the data.

The analysis remains limited to a single dataset and a single encoder architecture. To evaluate generalizability, future work should test these design choices across additional subject areas, item types, student populations, and model families, including decoder-based embedding models and larger generative models.

References

- Luca Benedetto, Giovanni Aradelli, Antonia Donvito, Alberto Lucchetti, Andrea Cappelli, and Paula Buttery. 2024. Using LLMs to simulate students' responses to exam questions. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11351–11368.
- Pedro Calais, Gabriel Franco, Themistoklis Nikas, Zilu Tang, Mark Crovella, Wagner Meira Jr, and Evimaria Terzi. 2024. Beyond accuracy: understanding the performance of llms on exams designed for humans.
- Radhika Kapoor, Sang T Truong, Nick Haber, Maria Araceli Ruiz-Primo, and Benjamin W Domingue. 2025. Prediction of item difficulty for reading comprehension items by creation of annotated item repository. *arXiv preprint arXiv:2502.20663*.
- John P Lalor, Hao Wu, and Hong Yu. 2019. Learning latent parameters without human response patterns: Item response theory with artificial crowds. 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)*, pages 4249–4259.
- Xinyi Lu and Xu Wang. 2024. Generative students: Using LLM-simulated student profiles to support question item evaluation. In *Proceedings of the Eleventh ACM conference on Learning @ Scale*, pages 16–27.
- Hotaka Maeda. 2025. Field-testing multiple-choice questions with ai examinees: English grammar items. *Educational and Psychological Measurement*, 85(2):221–244.
- Hotaka Maeda and Yikai Lu. 2026. Multimodal test item parameter prediction from text, images, and metadata: Fusing together ai vision and language models. *OSF Preprints*.
- Sungjin Nam. 2025. Domain-adaptive automated essay scoring with topic relevance learning. In *Proceedings of the Second Workshop on Automated Evaluation of Learning and Assessment Content, co-located with 26th International Conference on Artificial Intelligence in Education (AIED 2025)*.
- Christopher Ormerod. 2026. Reconstructing item characteristic curves using fine-tuned large language models. *arXiv preprint arXiv:2601.02580*.
- Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, and 1 others. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547.
- Kevin P Yancey, Andrew Runge, Geoffrey Laflair, and Phoebe Mulcaire. 2024. Bert-irt: Accelerating item piloting with bert embeddings and explainable irt models. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 428–438.

A Cross-Attention Results

With cross-attention, joint-parameter prediction with CRPS loss and passage-grouped batching produced the best average Pearson performance, mainly through stronger c recovery. In the CA condition, CRPS improved recovery for b (in both per- and joint-parameter settings) and for c (in the joint-parameter setting), while reducing a . Passage-grouped batching improved c but reduced a and b .

Cross-attention showed patterns similar to those in Section 5.1, improving c , reducing b , and producing mixed results for a . The single best c Pearson (0.484) came from CA:o with MSE and grouped batching, whereas the lowest RMSEs across all three parameters came from CA:x with grouped batching. The results indicate that CA can improve c recovery with trade-offs for other item parameters.

B Full Experimental Results

Table 3 reports the full results for all combinations of encoding, prediction, loss, batching, and cross-attention settings. The results show that the best-performing settings for each parameter varied across configurations, but the combination of separate-then-fuse encoding, joint-parameter prediction, CRPS loss, and passage-grouped batching consistently produced strong and balanced performance across all three parameters.

Bat.	Enc.	Loss	CA	Prd.	Pearson r ($\uparrow$)				RMSE ($\downarrow$)			
					a	b	c	Avg.	a	b	c	Avg.
R	C	MSE	x	P	0.532	0.432	0.204	0.389	0.291	0.854	0.075	0.407
			x	J	0.525	0.434	0.002	0.320	0.293	0.858	0.085	0.412
			x	P	0.514	0.445	0.189	0.383	0.302	0.849	0.075	0.409
			x	J	0.450	0.442	0.110	0.334	0.290	0.855	0.075	0.407
			o	P	0.533	0.433	0.333	0.433	0.282	0.870	0.075	0.409
			o	J	0.503	0.437	-0.024	0.305	0.288	0.859	0.076	0.408
	C	CRPS	x	P	0.524	0.432	0.231	0.396	0.277	0.861	0.075	0.404
			x	J	0.504	0.420	0.129	0.351	0.298	0.915	0.075	0.429
	SF		x	P	0.522	0.444	0.249	0.405	0.290	0.846	0.079	0.405
			x	J	0.529	0.427	0.226	0.394	0.289	0.895	0.079	0.421
			o	P	0.533	0.432	0.327	0.431	0.283	0.869	0.073	0.408
			o	J	0.537	0.428	0.302	0.422	0.275	0.877	0.075	0.409
PG	C	MSE	x	P	0.517	0.445	0.299	0.420	0.267	0.848	0.071	0.395
			x	J	0.531	0.442	0.273	0.415	0.266	0.851	0.071	0.396
			x	P	0.510	0.452	0.340	0.434	0.269	0.843	0.070	0.394
			x	J	0.517	0.444	0.269	0.410	0.270	0.848	0.072	0.397
			o	P	0.472	0.408	0.484	0.455	0.280	0.885	0.065	0.410
			o	J	0.510	0.403	0.275	0.396	0.274	0.890	0.072	0.412
	C	CRPS	x	P	0.522	0.445	0.306	0.424	0.270	0.851	0.072	0.398
			x	J	0.524	0.439	0.293	0.419	0.270	0.852	0.072	0.398
	SF		x	P	0.521	0.462	0.332	0.438	0.271	0.843	0.071	0.395
			x	J	0.524	0.456	0.338	0.439	0.270	0.844	0.071	0.395
			o	P	0.454	0.418	0.465	0.446	0.286	0.876	0.067	0.410
			o	J	0.491	0.410	0.466	0.456	0.278	0.876	0.067	0.407

Table 3: Results for the full combinations of batching (Bat.; R: Random, PG: Passage-Grouped), encoding (Enc.; C: Concatenated, SF: Separate-then-Fuse), cross-attention (CA; x: no, o: yes), loss function (MSE, CRPS), and prediction (Prd; P: per-parameter, J: joint) settings. Cross-attention was evaluated only with separate-then-fuse encoding.