

BOBCAT: Bilevel Optimization-Based Computerized Adaptive Testing

Aritra Ghosh and Andrew Lan
University of Massachusetts Amherst
{arighosh, andrewlan}@cs.umass.edu

Abstract

Computerized adaptive testing (CAT) refers to a form of tests that are personalized to every student/test taker. CAT methods adaptively select the next most informative question/item for each student given their responses to previous questions, effectively reducing test length. Existing CAT methods use item response theory (IRT) models to relate student ability to their responses to questions and static question selection algorithms designed to reduce the ability estimation error as quickly as possible; therefore, these algorithms cannot improve by learning from large-scale student response data. In this paper, we propose BOBCAT, a Bilevel Optimization-Based framework for CAT to directly learn a data-driven question selection algorithm from training data. BOBCAT is agnostic to the underlying student response model and is computationally efficient during the adaptive testing process. Through extensive experiments on five real-world student response datasets, we show that BOBCAT outperforms existing CAT methods (sometimes significantly) at reducing test length.

1 Introduction

One important feature of computerized/online learning platforms is computerized adaptive testing (CAT), which refers to tests that can accurately measure the ability/knowledge of a student/test taker using few questions/items, by using an algorithm to adaptively select the next question for each student given their response to previous questions [van der Linden and Glas, 2000; Luecht and Sireci, 2011]. An accurate and efficient estimate of a student’s knowledge levels helps computerized learning platforms to deliver personalized learning experiences for every learner.

A CAT system generally consists of the following components: an underlying psychometric model that links the question’s features and the student’s features to their response to the question, a bank of questions with features learned from prior data, and an algorithm that selects the next question for each student from the question bank and decides when to stop the test; see [Han, 2018] for an overview. Most commonly

used response models in CAT systems are item response theory (IRT) models, with their simplest form (1PL) given by

$$p(Y_{i,j} = 1) = \sigma(\theta_i - b_j), \quad (1)$$

where $Y_{i,j}$ is student i ’s binary-valued response to question j , where 1 denotes a correct answer, $\sigma(\cdot)$ is the sigmoid/logistic function, and $\theta_i \in \mathbb{R}$ and $b_j \in \mathbb{R}$ are scalars corresponding to the student’s ability and the question’s difficulty, respectively [Lord, 1980; Rasch, 1993]. More complex IRT models use additional question features such as the scale and guessing parameters or use multidimensional student features, i.e., their knowledge levels on multiple skills [Reckase, 2009].

Most commonly used question selection algorithms in CAT systems select the most informative question that minimizes the student feature measurement error; see [van der Linden and Pashley, 2009] for an overview. Specifically, in each step of the adaptive testing process (indexed by t) for student i , they select the next question as

$$j_i^{(t)} = \operatorname{argmax}_{j \in \Omega_i^{(t)}} I_j(\hat{\theta}_i^{(t-1)}), \quad (2)$$

where $\Omega_i^{(t)}$ is the set of available questions to select for this student at time step t (the selected question at each time step is removed afterwards), $\hat{\theta}_i^{(t-1)}$ is the current estimate of their ability parameter given previous responses $Y_{i,j_i^{(1)}}, \dots, Y_{i,j_i^{(t-1)}}$, and $I_j(\cdot)$ is the informativeness of question j . In the context of 1PL IRT models, most informativeness metrics will select the question with difficulty closest to the current estimate of the student’s ability, i.e., selecting the question that the student’s probability of answering correctly is closest to 50%. This criterion coincides with uncertainty sampling [Lewis and Gale, 1994] for binary classification, a commonly used method in active learning [Settles, 2012] that is deployed in real-world CAT systems [Settles *et al.*, 2020].

Despite the effectiveness of existing CAT methods, two limitations hinder their further improvement. First, most question selection algorithms are specifically designed for IRT models (1). The *highly structured* nature of IRT models enables theoretical characterization of question informativeness but limits their ability to capture complex student-question interactions compared to more flexible, deep neural network-based models [Cheng *et al.*, 2019; Wang *et al.*, 2020a]. This limitation is evident on large-scale student response datasets (often with millions of responses) that have

been made available [Choi *et al.*, 2020; Wang *et al.*, 2020b]. Second, most existing question selection algorithms are *static* since they require a predefined informativeness metric (2); they can only use large-scale student response data to improve the underlying IRT model (e.g., calibrating question difficulty parameters) but not the question selection algorithm. Therefore, they will not significantly improve over time as more students take tests. Recently, there are ideas on using reinforcement learning to learn question selection algorithms [Nurakhmetov, 2019; Li *et al.*, 2020]; however, these methods have not been validated on real data.

1.1 Contributions

In this paper, we propose BOBCAT, a Bilevel Optimization-Based framework for Computerized Adaptive Testing. BOBCAT is based on the key observation that the ultimate goal of CAT is to reduce test length. Therefore, estimating student ability parameters is a *proxy* of the real objective: *predicting a student's responses to all questions on a long test that cannot be feasibly administered*. We make three key contributions:

First, we recast CAT as a bilevel optimization problem [Franceschi *et al.*, 2018] in the meta learning [Finn *et al.*, 2017] setup: in the *outer-level* optimization problem, we learn both the response model parameters and a data-driven question selection algorithm by *explicitly* maximizing the predictive likelihood of student responses in a held-out *meta* question set. In the *inner-level* optimization problem, we adapt the outer-level response model to each student by maximizing the predicted likelihood of their responses in an observed *training* question set. This bilevel optimization framework directly learns an effective and efficient question selection algorithm through the training-meta setup. Moreover, BOBCAT is agnostic to the underlying response model, compatible with both IRT models and deep neural network-based models; Once learned, the question selection algorithm selects the next question from past question responses directly, without requiring the student parameters to be repeatedly estimated in real time during the CAT process.

Second, we employ a *biased* approximate estimator of the gradient w.r.t. the question selection algorithm parameters in the bilevel optimization problem. This approximation leverages the influence of each question on the algorithm parameters [Koh and Liang, 2017] to reduce the variance in the gradient estimate and leads to better question selection algorithms than an unbiased gradient estimator.

Third, we verify the effectiveness of BOBCAT through extensive quantitative and qualitative experiments on five large-scale, real-world student response datasets. We observe that the learned data-driven question selection algorithms outperform existing CAT algorithms at reducing test length, requiring 50% less questions to reach the same predictive accuracy on meta question set in some cases; this improvement is generally more significant on larger datasets. Our implementation will be publicly available at <https://github.com/arghosh/BOBCAT>.

Remarks. We emphasize that the goal of this paper is to learn data-driven question selection algorithms, **not** to develop the best underlying student response model. We also acknowledge that BOBCAT may not be directly applicable

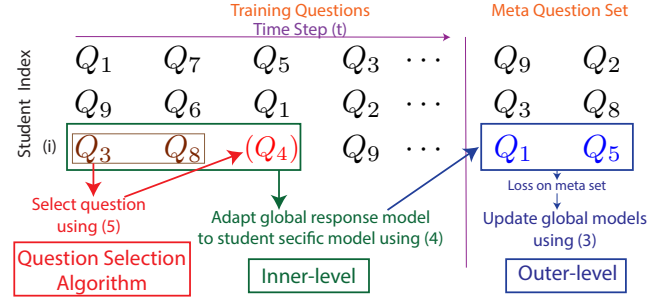


Figure 1: Overview of the BOBCAT framework.

to real-world CAT settings since its data-driven nature makes it hard to i) theoretically analyze and ii) prone to biases in historical data. Nevertheless, our promising results show that it is possible to improve the efficiency of CAT with large-scale student response data. Additionally, other important CAT aspects such as question exposure control [Veldkamp and van der Linden, 2008] need to be further investigated; we provide preliminary qualitative analyses in Section 3.1.

2 The BOBCAT Framework

We now detail the BOBCAT framework, visualized in Figure 1. Let N and Q denote the number of students and questions in the student response dataset we use to train BOBCAT, respectively. For a single student i , we sequentially select a total of n ($\ll |\Omega_i^{(1)}|$) questions¹, $\{j_i^{(1)}, \dots, j_i^{(n)}\}$, observe their responses, and predict their response on a held-out set of meta questions, Γ_i ; $\Omega_i^{(1)}$ denotes the initial set of available questions and $\Omega_i^{(1)} \cap \Gamma_i = \emptyset$. The training and meta question sets are randomly selected and not the same for each student in the dataset. We solve the following bilevel optimization problem [Franceschi *et al.*, 2018; Rajeswaran *et al.*, 2019]:

$$\underset{\gamma, \phi}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^N \sum_{j \in \Gamma_i} \ell(Y_{i,j}, g(j; \theta_i^*)) := \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\theta_i^*, \Gamma_i) \quad (3)$$

$$\text{s.t. } \theta_i^* = \underset{\theta_i}{\text{argmin}} \sum_{t=1}^n \ell(Y_{i,j_i^{(t)}}, g(j_i^{(t)}; \theta_i)) + \mathcal{R}(\gamma, \theta_i) := \mathcal{L}'(\theta_i) \quad (4)$$

$$\text{where } j_i^{(t)} \sim \Pi(Y_{i,j_i^{(1)}}, \dots, Y_{i,j_i^{(t-1)}}; \phi) \in \Omega_i^{(t)}. \quad (5)$$

Here, γ and ϕ are the *global* response model and question selection algorithm parameters, respectively. $g(\cdot)$ is the response model, which takes as input the index of the question of interest, j , and uses the *local* parameter specific to student i , θ_i^* , to output the prediction of the student's likelihood of responding to the question correctly. $\Pi(\cdot)$ is the question selection algorithm (red box in Figure 1), which takes as input the student's responses to previously selected questions and outputs the index of the next selected question.

¹BOBCAT is applicable to both fixed-length and variable-length CAT designs [Choi *et al.*, 2011]; we study the former in this paper.

The outer-level optimization problem (blue box in Figure 1) minimizes the binary cross-entropy loss, $\ell(\cdot)$, on the *meta* question sets across all students to learn both the global response model and the question selection algorithm; $\mathcal{L}(\cdot)$ corresponds to the sum of this loss over questions each student responded to in the meta question set. The inner-level optimization problem (green box in Figure 1) minimizes $\mathcal{L}'(\cdot)$, the cross-entropy loss on a small number of questions selected for each student on the *training* question set to adapt the global response model to each local student, resulting in a student-specific parameter θ_i^* ; $\mathcal{R}(\gamma, \theta_i)$ is a regularization term that penalizes large deviations of the local parameters from their global values. Note that θ_i^* is a function of the global parameters γ and ϕ , reflected through both the regularization term in (4) and the question selection algorithm through questions it selects for this student in (5).

Response Model. The response model $g(\cdot)$ can be taken as either IRT models or neural network-based models. In the case of IRT models, the global parameters γ corresponds to the combination of the question difficulties and the student ability prior. We adapt these parameters to each local student through their responses to selected questions in the inner-level optimization problem. In our experiments, we only use the global student ability as the prior mean of each local student’s ability estimate and keep the question difficulties fixed in the inner-level optimization problem, following the typical setup in real-world CAT systems. In the case of neural network-based models, the parameters are usually not associated with any specific meaning; following standard practices in meta-learning [Lee *et al.*, 2019], we fix part of the network (e.g., all weights and biases, which one can regard as a non-linear version of question difficulties) and optimize the rest of the network (e.g., the input vector, which one can regard as student abilities) in the inner-level optimization problem.

Question Selection Algorithm. The question selection algorithm $\Pi(\cdot)$ can be either deterministic or probabilistic, i.e., it either outputs a single selected question or a probability distribution over available questions. We define the input *state* vector to the question selection algorithm at step t as $\mathbf{x}_i^{(t)} \in \{-1, 0, 1\}^Q$, where an entry of -1 denotes an incorrect response to a past selected question, 1 denotes a correct response, while 0 denotes questions that have not been selected. We do not include the time step at which a question is selected in the state vector since in CAT settings, the student’s true ability is assumed to be static during the testing process while an estimate is being updated. Although any differentiable model architecture can be used for the question selection algorithm, we use the multi-layer perceptron model that is invariant to question ordering. For probabilistic question selection algorithms, we select a question by sampling from the output distribution $j_i^{(t)} \sim \Pi(\mathbf{x}_i^{(t)}, \Omega_i^{(t)}; \phi)$.

2.1 Optimization

We use gradient descent (GD) to solve the inner-level optimization problem for the local response model parameters θ_i^* , following model-agnostic meta learning [Finn *et al.*, 2017]. In particular, we let the local student-specific parameter deviate from the global response model parameters by taking K

GD steps from γ , where each step is given as

$$\theta_i \leftarrow \theta_i - \alpha \nabla_{\theta} \sum_{t=1}^n \ell(Y_{i, j_i^{(t)}}, g(j_i^{(t)}; \theta)) \Big|_{\theta_i}, \quad (6)$$

where α is the learning rate. We do not explicitly use regularization for GD steps since early stopping (with only a few GD steps) is equivalent to a form of regularization [Rajeswaran *et al.*, 2019]. Computing the gradient w.r.t. the global parameters γ requires us to compute the gradient w.r.t. the gradient in the inner-level optimization problem in (6) (also referred to as the meta-gradient), which can be computed using automatic differentiation [Paszke *et al.*, 2017]. Computing the exact meta-gradient requires second-order derivatives; however, we found that first-order approximation works well in practice and leads to low computational complexity. Similarly, to learn the selection algorithm parameters ϕ , we need to compute the gradient of the outer-level objective in (3) w.r.t. ϕ through the student-specific parameters $\theta_i^*(\gamma, \phi)$, i.e., the solution to the inner-level optimization problem. The gradient for a single student i (the full gradient sums across all students) is given by

$$\nabla_{\phi} \mathcal{L}(\theta_i^*(\gamma, \phi), \Gamma_i) = \nabla_{\phi} \mathbb{E}_{j_i^{(1:n)} \sim \Pi(\cdot; \phi)} [\mathcal{L}(\theta_i^*(\gamma, \{j_i^{(1:n)}\}), \Gamma_i)], \quad (7)$$

where we replace the dependence of θ_i^* on the parameters of the question selection algorithm, ϕ , with the indices of the selected questions, $j_i^{(1:n)}$, which we need to backpropagate through. The discrete nature of these variables makes them non-differentiable so that we cannot compute the exact gradient. Next, we will detail two ways to estimate this gradient.

Unbiased Gradient Estimate

We can use the score function-based identity ($\frac{\partial \log f(X; \phi)}{\partial \phi} = \frac{\partial f(X; \phi) / \partial \phi}{f(X; \phi)}$ for any probability distribution $f(X; \phi)$) to estimate the unbiased gradient in (7) [Williams, 1992] as

$$\begin{aligned} & \nabla_{\phi} \mathbb{E}_{j_i^{(1:n)} \sim \Pi(\cdot; \phi)} [\mathcal{L}(\theta_i^*(\gamma, \{j_i^{(1:n)}\}), \Gamma_i)] \\ &= \mathbb{E}_{j_i^{(1:n)} \sim \Pi(\cdot; \phi)} \left[(\mathcal{L}(\theta_i^*, \Gamma_i) - b_i) \nabla_{\phi} \log \prod_{t=1}^n \Pi(j_i^{(t)} | \mathbf{x}_i^{(t)}; \phi) \right] \\ &= \mathbb{E}_{j_i^{(1:n)} \sim \Pi(\cdot; \phi)} \left[(\mathcal{L}(\theta_i^*, \Gamma_i) - b_i) \sum_{t=1}^n \nabla_{\phi} \log \Pi(j_i^{(t)} | \mathbf{x}_i^{(t)}; \phi) \right], \end{aligned} \quad (8)$$

where b_i is a control variable to the reduce the variance of the gradient estimate. This unbiased gradient resembles reinforcement learning-type algorithms for CAT, an idea discussed in [Nurakhmetov, 2019]. We use proximal policy optimization for its training stability with an actor network and a critic network [Schulman *et al.*, 2017]; we provide details of these networks in the supplementary material to be released in the full version of the paper.

We observe that this unbiased gradient estimate updates the question selection algorithm parameters through the selected questions only, without including observations on the available but not selected questions, resulting in slow empirical convergence in practice. However, incorporating information on unselected questions into the gradient computation may

lead to lower variance in the gradient and stabilize the training process. Next, we detail a biased approximation to the gradient using all the available training questions.

Approximate Gradient Estimate

We can rewrite the gradient in (7) as

$$\nabla_{\phi} \mathcal{L}(\theta_i^*(\gamma, \phi), \Gamma_i) = \nabla_{\theta_i^*} \mathcal{L}(\theta_i^*, \Gamma_i) \nabla_{\phi} \theta_i^*(\gamma, \phi). \quad (9)$$

The gradient w.r.t. θ_i^* can be computed exactly; next, we discuss the computation of $\nabla_{\phi} \theta_i^*(\gamma, \phi)$ in detail for a single time step t . We can rewrite the inner-level optimization in (4) by splitting the current question index $j_i^{(t)}$ from previously selected question indices $j_i^{(1)}, \dots, j_i^{(t-1)}$ as

$$\begin{aligned} \theta_i^* = \operatorname{argmin}_{\theta_i} \sum_{\tau=1}^{t-1} \ell(Y_{i,j_i^{(\tau)}}, g(j_i^{(\tau)}; \theta_i)) + \mathcal{R}(\gamma, \theta_i) \\ + \sum_{j \in \Omega_i^{(t)}} w_j(\phi) \ell(Y_{i,j}, g(j; \theta_i)), \end{aligned} \quad (10)$$

where $w_j(\phi) = 1$ if $j = j_i^{(t)}$ and $w_j(\phi) = 0$ for all other available questions. In (10), we can compute the derivative $\frac{d\theta_i^*}{dw_j(\phi)}$ for all available question indices in $\Omega_i^{(t)}$ regardless of whether they are selected at time step t , using the implicit function theorem [Cook and Weisberg, 1982] as

$$\frac{d\theta_i^*}{dw_j(\phi)} = -\left(\nabla_{\theta_i}^2 \mathcal{L}'\right)^{-1} \nabla_{\theta_i} \ell(Y_{i,j}, g(j; \theta_i)) \Big|_{\theta_i^*}.$$

This gradient can be computed without explicitly computing the inverse Hessian matrix using automatic differentiation in a way similar to that for the global response model parameters γ . However, we still need to compute $\frac{\partial w_j(\phi)}{\partial \Pi(j|\mathbf{x}_i^{(t)}; \phi)}$, which is not differentiable; since $w_j(\phi) = \Pi(j|\mathbf{x}_i^{(t)}; \phi)$ holds when the selection algorithm network puts all the probability mass on a single question, we can use the approximation $w_j(\phi) \approx \Pi(j|\mathbf{x}_i^{(t)}; \phi)$. From (9) and (10), it turns out that under this approximation, the full gradient with respect to a single question, $\frac{\partial \mathcal{L}(\theta_i^*, \Gamma_i)}{\partial \Pi(j|\mathbf{x}_i^{(t)}; \phi)}$, is the widely used influence function score [Koh and Liang, 2017]:

$$-\nabla_{\theta_i} \mathcal{L}(\theta_i, \Gamma_i) \left(\nabla_{\theta_i}^2 \mathcal{L}' \right)^{-1} \nabla_{\theta_i} \ell(Y_{i,j}, g(j; \theta_i)) \Big|_{\theta_i^*} := \mathcal{I}_i(j), \quad (11)$$

where $\mathcal{I}_i(j)$, the influence function score of question j , computes the change in the loss on the meta question set under small perturbations in the weight of this question, $w_j(\phi)$, in (10). Intuitively, we would want to select available training questions with gradients that are similar to the gradient on the meta question set, i.e., those with the most information on meta questions; the approximation enables us to learn such a question selection algorithm by backpropagating the influence score as gradients through all available questions in the training question set. In contrast, for the unbiased gradient in (8), $\frac{\partial \mathcal{L}(\theta_i^*, \Gamma_i)}{\partial \Pi(j|\mathbf{x}_i^{(t)}; \phi)}$ equals zero for all unselected questions

Algorithm 1 BOBCAT training process

- 1: Initialize global parameters γ, ϕ , learning rates η_1, η_2, α , and number of GD steps at the inner-level, K .
- 2: **while** not converged **do**
- 3: Randomly sample a mini-batch of students \mathcal{B} with training and meta question sets $\{\Omega_i^{(1)}, \Gamma_i\}_{i \in \mathcal{B}}$.
- 4: **for** $t \in 1 \dots n$ **do**
- 5: Encode the student's current state $\mathbf{x}_i^{(t)}$ based on their responses to previously selected questions.
- 6: Select question $j_i^{(t)} \sim \Pi(\mathbf{x}_i^{(t)}; \phi)$ for each student.
- 7: Optimize θ_i^* in Eq. 6 using learning rate α and K GD steps on observed responses $\{Y_{i,j_i^{(1:t)}}\}$.
- 8: Estimate the unbiased (or the approximate) gradient $\nabla_{\phi} \mathcal{L}(\theta_i^*, \Gamma_i)$ using Eq. 8 (or Eq. 11).
- 9: Update ϕ : $\phi \leftarrow \phi - \frac{\eta_2}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_{\phi} \mathcal{L}(\theta_i^*, \Gamma_i)$.
- 10: **end for**
- 11: Update γ : $\gamma \leftarrow \gamma - \frac{\eta_1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_{\gamma} \mathcal{L}(\theta_i^*(\gamma, \phi), \Gamma_i)$.
- 12: **end while**

and equals $-(\mathcal{L}(\theta_i^*, \Gamma_i) - b_i) \log \Pi(j_i^{(t)}|\mathbf{x}_i^{(t)}; \phi)$ for the selected question $j_i^{(t)}$. This biased approximation (often known as the straight-through estimator) has been successfully applied in previous research for neural network quantization and leads to lower empirical variance [Bengio *et al.*, 2013]. Algorithm 1 summarizes BOBCAT's training process.

Computational Complexity. At training time, we need to solve the full BOBCAT bilevel optimization problem, which is computationally intensive on large datasets. However, at test time, when we need to select the next question for each student, we only need to use their past responses as input to the learned question selection algorithm $\Pi(\cdot; \phi)$ to get the selected question as output; this operation is more computationally efficient than existing CAT methods that require updates to the student's ability estimate after every question.

3 Experiments

We now detail both quantitative and qualitative experiments we conducted on five real-world student response datasets to validate BOBCAT's effectiveness.

Datasets, Training, Testing and Evaluation Metric. We use five publicly available benchmark datasets: EdNet², Junyi³, Eedi-1, Eedi-2⁴, and ASSISTments⁵. In Table 3, we list the number of students, the number of questions, and the number of interactions. We provide preprocessing details and additional background on each dataset in the supplementary material. We perform 5-fold cross validation for all datasets; for each fold, we use 60%-20%-20% *students* for training, validation, and testing, respectively. For each

²<https://github.com/riiid/ednet>

³<https://www.kaggle.com/junyiacademy/learning-activity-public-dataset-by-junyi-academy>

⁴<https://eedi.com/projects/neurips-education-challenge>

⁵<https://sites.google.com/site/assistmentsdata/home/assistent-2009-2010-data>

Dataset	n	IRT-Active	BiIRT-Active	BiIRT-Unbiased	BiIRT-Approx	BiNN-Approx
EdNet	1	70.08	70.92	71.12	71.22	71.22
	3	70.63	71.16	71.3	71.72	71.82
	5	71.03	71.37	71.45	71.95	72.17
	10	71.62	71.75	71.79	72.33	72.55
Junyi	1	74.52	74.93	74.97	75.11	75.1
	3	75.19	75.48	75.53	75.76	75.83
	5	75.64	75.79	75.75	76.11	76.19
	10	76.27	76.28	76.19	76.49	76.62
Eedi-1	1	66.92	68.22	68.61	68.82	68.78
	3	68.79	69.45	69.81	70.3	70.45
	5	70.15	70.28	70.47	70.93	71.37
	10	71.72	71.45	71.57	72.0	72.33
Eedi-2	1	63.75	64.83	65.22	65.3	65.65
	3	65.25	66.42	67.09	67.23	67.79
	5	66.41	67.35	67.91	68.23	68.82
	10	68.04	68.99	68.84	69.47	70.04
ASSISTments	1	66.19	68.69	69.03	69.17	68.0
	3	68.75	69.54	69.78	70.21	68.73
	5	69.87	69.79	70.3	70.41	69.03
	10	71.04	70.66	71.17	71.14	69.75

Table 1: Average predictive accuracy on the meta question set across folds on all datasets. Best methods are shown in **bold** font. For standard deviations and results on all methods, refer to Figure 2 and Tables in the supplementary material.

fold, we use the validation students to perform early stopping and tune the parameters for every method. For BOBCAT, we partition the questions responded to by each student into the training ($\Omega_i^{(1)}$, 80%) and meta (Γ_i , 20%) question sets. To prevent overfitting, we randomly generate these partitions in each training epoch. We use both accuracy and the area under the receiver operating characteristics curve (AUC) as metrics to evaluate the performance of all methods on predicting binary-valued student responses on the meta set Γ_i . We implement all methods in PyTorch and run our experiments in a NVIDIA TitanX/1080Ti GPU.

Methods and Baselines. For existing CAT methods, we use **IRT-Active**, the uncertainty sampling-based [Lewis and Gale, 1994] active learning question selection algorithm, which selects the next question with difficulty closest to a student’s current ability estimate, as a baseline [Settles *et al.*, 2020]. This method coincides with the question information-based CAT methods under the IPL IRT model. We also use an additional baseline that selects the next question randomly, which we dub **IRT-Random**. For BOBCAT, we consider the cases of using IRT models (which we dub as **BiIRT**) and neural networks (which we dub as **BiNN**) as the response model. For both BiIRT and BiNN, we use four question selection algorithms: in addition to the **-Active** and **-Random** algorithms above, we also use learned algorithms with the **-Unbiased** gradient (8) and the approximate (**-Approx**) gradient (11) on the question selection algorithm parameters ϕ .

Networks and Hyper-parameters. We train IRT models using logistic regression with l_2 -norm regularization. For IRT-Active, we compute the student’s current ability estimate with l_2 -norm regularization to penalize deviation from the mean student ability parameter. For BiNN, we use a two-layer, fully-connected network (with 256 hidden nodes, ReLU nonlinearity, 20% dropout rate, and a final sigmoid out-

Dataset	n	IRT-Active	BiIRT-Active	BiIRT-Unbiased	BiIRT-Approx	BiNN-Approx
EdNet	1	73.58	73.82	74.14	74.34	74.41
	3	74.14	74.21	74.49	75.26	75.43
	5	74.6	74.56	74.77	75.68	76.07
	10	75.35	75.21	75.39	76.35	76.74
Junyi	1	74.92	75.53	75.67	75.91	75.9
	3	76.06	76.52	76.71	77.11	77.16
	5	76.82	77.07	77.07	77.69	77.8
	10	77.95	77.95	77.86	78.45	78.6
Eedi-1	1	68.02	70.22	70.95	71.34	71.33
	3	71.63	72.47	73.26	74.21	74.44
	5	73.69	73.97	74.54	75.47	76.0
	10	76.12	75.9	76.34	77.07	77.51
Eedi-2	1	69.0	70.15	70.64	70.81	71.24
	3	71.11	72.18	73.11	73.37	73.88
	5	72.42	73.21	74.19	74.55	75.2
	10	74.36	75.17	75.37	75.96	76.63
ASSISTments	1	69.14	70.55	71.0	71.33	70.12
	3	71.17	71.6	72.35	73.16	71.57
	5	72.26	71.65	73.1	73.71	72.14
	10	73.62	72.52	74.38	74.66	73.59

Table 2: Average AUC on the meta question set across folds on all datasets. For standard deviations and results on all methods, refer to Figures and Tables in the supplementary material.

Dataset	EdNet	Junyi	Eedi-1	Eedi-2	ASSISTments
Students	312K	52K	119K	5K	2.3K
Questions	13K	25.8K	27.6K	1K	26.7K
Interactions	76M	13M	15M	1.4M	325K

Table 3: Dataset statistics.

put layer) [Goodfellow *et al.*, 2016] as the response model, with a student-specific, 256-dimensional ability vector as input. We use another fully-connected network (with two hidden layers, 256 hidden nodes, Tanh nonlinearity, and a final softmax output layer) [Goodfellow *et al.*, 2016] as the question selection algorithm. For BiNN/IRT-Unbiased, we use another fully-connected critic network (two hidden layers, 256 hidden nodes, Tanh nonlinearity) in addition to the question selection actor network. For BiIRT and BiNN, we learn the global response model parameters γ and question selection algorithm parameters ϕ using the Adam optimizer [Kingma and Ba, 2015] and learn the response parameters adapted to each student (in the inner-level optimization problem) using the SGD optimizer [Goodfellow *et al.*, 2016]. We provide specific hyper-parameter choices and batch sizes for each dataset in the supplementary material. For all methods, we select $n \in \{1, 3, 5, 10\}$ questions for each student.

3.1 Results and Discussion

In Table 1, we list the mean accuracy numbers across all folds for selected BOBCAT variants and IRT-Active on all datasets; in Table 2, we do the same using the AUC metric. In the supplementary material, we provide results for all methods mentioned above and also list the standard deviations across folds. Using a neural network-based response model, BiNN-Approx outperforms other methods in most cases. Using an IRT response model, BiIRT-Approx performs similarly to BiNN-Approx and outperforms other methods. All BOBCAT vari-

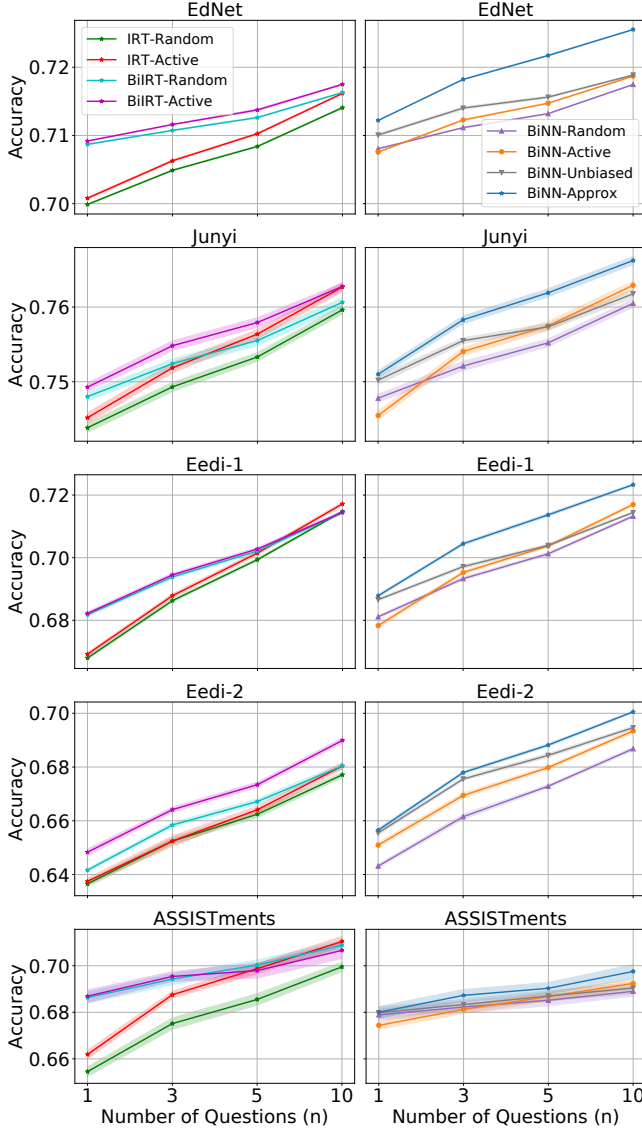


Figure 2: Average accuracy (dark lines) and 5-fold standard deviation (light fill lines) on all datasets. First column compares IRT vs BiIRT models; second column compares all BiNN models.

ants significantly outperform IRT-Active, which uses a static question selection algorithm. On the ASSISTments dataset, the smallest of the five, BiIRT-Approx outperforms BiNN-Approx, which overfits. These results show that i) BOBCAT improves existing CAT methods by explicitly learning a question selection algorithm from data, where the improvement is more obvious on larger datasets, and ii) since BOBCAT is agnostic to the underlying response model, one can freely choose either IRT models when training data is limited or neural network-based models when there is plenty of training data.

In Figure 2, we use a series of plots as ablation studies to present a more detailed comparison between different methods; here, we include random question selection as a bottom line. In the first column, we plot the mean and the standard

deviation of accuracy for IRT-Random, IRT-Active, BiIRT-Random, and BiIRT-Active versus the number of questions selected. On the Eedi-1 dataset, BiIRT-Active performs better than IRT-Active on smaller n but performs slightly worse for $n = 10$. On the ASSISTments dataset, we observe a high standard deviation for larger n ; nevertheless, BiIRT variants outperform IRT counterparts. On all other datasets, the BiIRT methods outperform their IRT counterparts significantly. To reach the same accuracy, on the EdNet, Eedi-2, and Junyi datasets, BiIRT-Active requires $\sim 30\%$ less questions compared to IRT-Active. This head-to-head comparison using IRT as the underlying response model demonstrates the power of bilevel optimization; even using static question selection algorithms, explicitly maximizing the predictive accuracy on a meta question set results in better performance, although the performance gain may not be significant.

In the second column, we compare different BOBCAT variants using the same underlying neural network-based response model. We observe that on all datasets, BiNN-Approx significantly outperforms other methods, reaching the same accuracy as BiNN-Active with 50%-75% less questions. This performance gain is more significant on larger datasets. It also significantly outperforms the unbiased gradient estimate, reaching the same accuracy with 10%-70% less questions. BiNN-Unbiased significantly outperforms BiNN-Active for smaller n but not for large n ; we believe the large variance of the unbiased gradient might be the reason for this behavior. This head-to-head comparison shows that our approximate gradient estimate stabilizes the model training process and leads to better model fit. Moreover, data-driven question selection algorithms learned through bilevel optimization are much better than standard static CAT question selection algorithms and get better with more training data.

Study: Ability Estimation. The goal of existing real-world CAT systems is to accurately estimate the student ability parameter under IRT models, which is then used for scoring. Therefore, we conduct an additional experiment on the Eedi-2 dataset using the squared error between the current ability parameter estimate $\hat{\theta}_i^{(n)}$ and the true ability θ_i as the evaluation metric. Since the true student ability is unknown in real student response datasets, we use the ability value estimated from all questions each student responded to as a substitute. We compare two methods: IRT-Active, with the underlying 1PL IRT model trained on the data and BiIRT-Approx, where we only use the learned model-agnostic question selection algorithm for evaluation in the setting of existing CAT methods. Figure 3(left) shows the ability estimation error (averaged over five folds) for different numbers of questions selected, n . We see that even though the goal of BiIRT-Approx is not ability parameter estimation, it is more effective than IRT-Active and can reach the same accuracy using up to 30% less questions, significantly reducing test length. Figure 3(right) shows the same comparison for models trained on 25% and 50% of the training data set. We see that BOBCAT can improve significantly as more training data becomes available while existing CAT methods cannot.

Study: Question Exposure and Content Overlap. In Table 4, we provide summary statistics on the question expo-

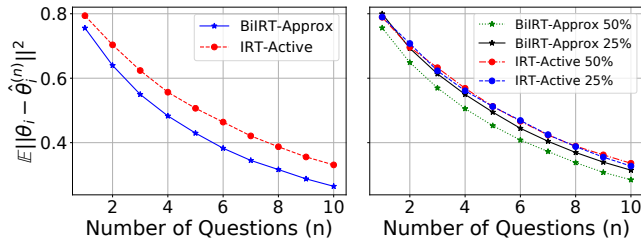


Figure 3: Ability estimation accuracy on the Eedi-2 dataset.

Method	Exposure (median)	Exposure (> 20%)	Overlap (mean)
IRT-Active	0.51%	0.25%	6.03%
BiNN-Approx	0%	1.54%	28.64%

Table 4: Question exposure and test overlap rates for the IRT-Active and BiNN-Approx methods on the Eedi-2 dataset.

sure rate (proportion of times a question was selected) and the test overlap rate (overlap among questions selected for two students) on the Eedi-2 dataset. We see that BOBCAT results in a slightly higher question exposure rate than existing CAT methods but still leads to an acceptable portion of overexposed questions (more than the recommended limit of 20% [Veldkamp and van der Linden, 2008]). However, BOBCAT results in a much higher test overlap rate than existing CAT methods [Stocking, 1994]. The reason for this observation is that BOBCAT favors a small subset of questions that are highly predictive of student responses to other questions, which we explain in detail next. Therefore, it is important for future work to develop constrained versions of BOBCAT to minimize its question exposure and test overlap rates before deployment.

Study: Question Selection. To gain deeper insights on why BOBCAT leads to more accurate question prediction but higher question exposure and test overlap rates, we conduct a qualitative investigation. One possible explanation is that the gradient update in (11) results in higher weights for questions that are more similar to the questions in the meta set (that are randomly selected out of all questions). Therefore, the BiIRT/NN-Approx methods favor questions that are highly representative of the entire set of questions, resulting in a higher test overlap rate in Table 4. We investigate the questions selected by the IRT-Active, BiIRT-Approx, and BiNN-Approx methods on the Eedi-2 dataset. We compute the weighted mutual information (MI) between each question and all others questions. We then use this score to assign each question to an ordered bin (from 1 to 10) according to their MI such that each bin has equally many questions. Specifically (with the following notations that apply to this study only), let i_1, \dots, i_m be the set of m students who responded to both questions j and k . The responses to questions j and k are denoted as $Y_{i_1,j}, \dots, Y_{i_m,j}$ and $Y_{i_1,k}, \dots, Y_{i_m,k}$. The mutual information $MI(j, k)$ between questions j and k is computed as

$$\sum_{x \in \{0,1\}} \sum_{y \in \{0,1\}} \frac{|Y_{i,j}=x, Y_{i,k}=y|}{m} \log \frac{m|Y_{i,j}=x, Y_{i,k}=y|}{|Y_{i,j}=x||Y_{i,k}=y|}.$$

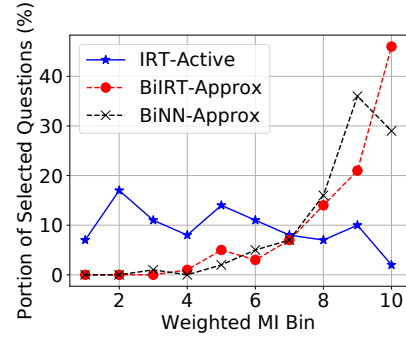


Figure 4: The relationship between questions selected by each method and the mutual information between them and all other questions in the Eedi-2 dataset. BOBCAT tends to select questions that are more informative.

We also compute the empirical frequency, p_j , of each question in the dataset, which is simply the portion of students that have responded to this question. The weighted MI of a single question j is then given by $\sum_{k, k \neq j} p_k MI(j, k)$. Intuitively, if the MI between a question and other questions in the dataset is high, i.e., in a higher MI bin, the question provides more information about each student than other questions. In Figure 4, we plot the fraction of questions selected by the IRT-Active, BiIRT-Approx, and BiNN-Approx method from each bin for students in the test set. These results confirms our intuition that the BiIRT-Approx and the BiNN-Approx methods favor questions that provide more information on all other questions for every student. On the contrary, we do not observe such trends for the IRT-Active method; the selected questions are evenly distributed in each bin. These trends explain why we observe lower question exposure and test overlap rates for the IRT-Active method in Table 4. Therefore, accurately predicting student responses to questions on a long test and selecting diverse questions to minimize question exposure and test overlap rates are conflicting objectives that need to be balanced. This observation further highlights the need to develop constrained versions of BOBCAT that can balance these objectives.

4 Conclusions and Future Work

In this paper, we proposed BOBCAT, a bilevel optimization framework for CAT, which is agnostic of the underlying student response model and learns a question selection algorithm from training data. Through extensive experiments on five real-world student response datasets, we demonstrated that BOBCAT can significantly outperform existing CAT methods at reducing test length. Avenues of future work include i) incorporating question exposure and content balancing constraints [Kingsbury and Zara, 1991] into BOBCAT to make it deployable in real-world tests and ii) studying the fairness aspects of BOBCAT due to potential biases in training data.

Acknowledgements

We thank the National Science Foundation for their support under grant IIS-1917713 and Stephen Sireci for helpful discussions.

References

- [Bengio *et al.*, 2013] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.
- [Cheng *et al.*, 2019] Song Cheng, Qi Liu, Enhong Chen, Zai Huang, Zhenya Huang, Yiyang Chen, Haiping Ma, and Guoping Hu. Dirt: Deep learning enhanced item response theory for cognitive diagnosis. In *International Conference on Information and Knowledge Management*, pages 2397–2400, 2019.
- [Choi *et al.*, 2011] Seung W Choi, Matthew W Grady, and Barbara G Dodd. A new stopping rule for computerized adaptive testing. *Educational and Psychological Measurement*, 71(1):37–53, 2011.
- [Choi *et al.*, 2020] Youngduck Choi, Youngnam Lee, Dongmin Shin, Junghyun Cho, Seoyon Park, Seewoo Lee, Jineon Baek, Chan Bae, Byungsoo Kim, and Jaewe Heo. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer, 2020.
- [Cook and Weisberg, 1982] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.
- [Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, volume 70, pages 1126–1135, 2017.
- [Franceschi *et al.*, 2018] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pages 1568–1577, 2018.
- [Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [Han, 2018] Kyung Chris Tyek Han. Components of the item selection algorithm in computerized adaptive testing. *Journal of Educational Evaluation for Health Professions*, 15, 2018.
- [Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representations*, May 2015.
- [Kingsbury and Zara, 1991] C Gage Kingsbury and Anthony R Zara. A comparison of procedures for content-sensitive item selection in computerized adaptive tests. *Applied measurement in education*, 4(3):241–261, 1991.
- [Koh and Liang, 2017] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894, 2017.
- [Lee *et al.*, 2019] Kwonjoon Lee, Subhansu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019.
- [Lewis and Gale, 1994] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, July 1994.
- [Li *et al.*, 2020] Xiao Li, Hanchen Xu, Jinming Zhang, and Hua-hua Chang. Deep reinforcement learning for adaptive learning systems. *arXiv preprint arXiv:2004.08410*, 2020.
- [Lord, 1980] Frederick Lord. *Applications of Item Response Theory to Practical Testing Problems*. Erlbaum Associates, 1980.
- [Luecht and Sireci, 2011] Richard M Luecht and Stephen G Sireci. A review of models for computer-based testing. research report 2011-12. *College Board*, 2011.
- [Nurakhmetov, 2019] Darkhan Nurakhmetov. Reinforcement learning applied to adaptive classification testing. In *Theoretical and Practical Advances in Computer-based Educational Measurement*, pages 325–336. Springer, Cham, 2019.
- [Paszke *et al.*, 2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS Workshop on Autodiff*, 2017.
- [Rajeswaran *et al.*, 2019] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pages 113–124, 2019.
- [Rasch, 1993] Georg Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests*. MESA Press, 1993.
- [Reckase, 2009] Mark D Reckase. *Multidimensional item response theory models*. Springer, 2009.
- [Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [Settles *et al.*, 2020] Burr Settles, Geoffrey T. LaFlair, and Masato Hagiwara. Machine learning-driven language assessment. *Transactions of the Association for Computational Linguistics*, 8:247–263, 2020.
- [Settles, 2012] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, Nov. 2012.
- [Stocking, 1994] Martha L Stocking. Three practical issues for modern adaptive testing item pools 1. *ETS Research Report Series*, 1994(1):i–34, 1994.
- [van der Linden and Glas, 2000] Wim J van der Linden and Cees AW Glas. *Computerized adaptive testing: Theory and practice*. Springer, 2000.
- [van der Linden and Pashley, 2009] Wim J van der Linden and Peter J Pashley. Item selection and ability estimation in adaptive testing. In *Elements of adaptive testing*, pages 3–30. Springer, 2009.
- [Veldkamp and van der Linden, 2008] Bernard P Veldkamp and Wim J van der Linden. Implementing sympon-hetter item-exposure control in a shadow-test approach to constrained adaptive testing. *International Journal of Testing*, 8(3):272–289, 2008.
- [Wang *et al.*, 2020a] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. Neural cognitive diagnosis for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6153–6161, 2020.
- [Wang *et al.*, 2020b] Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Yordan Zaykov, José Miguel Hernández-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton, Simon Peyton Jones, et al. Diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061*, 2020.
- [Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Supplementary Material

5 Unbiased Gradient Full Objective

We will use the term *reward* to denote the negative loss value $-\mathcal{L}$. In (8), the gradient is computed based on the loss on the meta question set and the selected question probability. In practice, we can use an actor-critic network, where the actor network computes the question selection probabilities $\Pi(\cdot|\mathbf{x}_i^{(t)}; \phi)$ and the critic network predicts a scalar expected reward value $V(\mathbf{x}_i^{(t)})$. The advantage value for the selected question $j_i^{(t)}$ is defined as,

$$A(\mathbf{x}_i^{(t)}, j_i^{(t)}) = -(\mathcal{L}(\theta_i^*, \Gamma_i) - b_i) - V(\mathbf{x}_i^{(t)}).$$

The actor network updates the network ϕ to improve the reward (or equivalently the advantage value). We use proximal policy optimization (PPO) for learning the unbiased model. The PPO model, a off-policy policy gradient method, takes multiple gradient steps on the parameter ϕ . Since, after the first step, the updated model ϕ is different than the model ϕ_{old} , used to compute the reward, we need to adjust the importance weights to get the unbiased gradient [Schulman *et al.*, 2017]. The proximal policy optimization objective at time step t for the actor network is,

$$L_1 = -\mathbb{E}_{j_i^{(t)} \sim \Pi(\phi_{\text{old}})} \left[\min \left\{ \frac{\Pi(j_i^{(t)}|\mathbf{x}_i^{(t)}; \phi)}{\Pi(j_i^{(t)}|\mathbf{x}_i^{(t)}; \phi_{\text{old}})} A(\mathbf{x}_i^{(t)}, j_i^{(t)}), \right. \right. \\ \left. \left. \text{Clip} \left\{ \frac{\Pi(j_i^{(t)}|\mathbf{x}_i^{(t)}; \phi)}{\Pi(j_i^{(t)}|\mathbf{x}_i^{(t)}; \phi_{\text{old}})}, 1 - \epsilon, 1 + \epsilon \right\} A(\mathbf{x}_i^{(t)}, j_i^{(t)}) \right\} \right], \quad (12)$$

where the function $\text{Clip}(r, 1 - \epsilon, 1 + \epsilon)$ returns $\min\{1 + \epsilon, \max\{r, 1 - \epsilon\}\}$; this constraint stabilizes the network by not taking large gradient steps. In addition, to encourage exploration, PPO adds an entropy objective based on the actor output probability distribution,

$$L_2 = \sum_{j \in \Omega_i^{(t)}} \left[\Pi(j|\mathbf{x}, \phi) \log \Pi(j|\mathbf{x}_i^{(t)}, \phi) \right]. \quad (13)$$

The critic network updates the parameter based on the MSE loss between the expected reward and the true reward,

$$L_3 = \|V(\mathbf{x}_i^{(t)}) + (\mathcal{L}(\theta_i^*, \Gamma_i) - b_i)\|^2. \quad (14)$$

We compute the reward for a student based on the accuracy on the meta question set. Moreover, we observed that the performance of each student differs considerably and the widely-used single moving average baseline does not perform that well; thus, we decided to use a different baseline for each student. We use meta-set performance based on a random selection algorithm as the baseline for each student. The baseline computation increases the computational complexity; however, the critic network output $V(\mathbf{x}_i^{(t)})$ only needs to predict how good the actor network is compared to a random selection algorithm. This choice of baseline worked well in all of our experiments. The final objective of PPO uses a weighted combination of these individual loss terms $(L_1 + 0.01L_2 + 0.5L_3)$.

Dataset	Students	Questions	Interactions	R
EdNet	312,372 (~0.8M)	13,169	76,489,425 (~95M)	1
Junyi	52224 (72,758)	25,785	13,603,481 (~16M)	2
Eedi-1	118,971	27,613	15,867,850	2
Eedi-2	4,918	948	1,382,727	5
ASSISTments	2,313 (4,217)	26,688	325,359 (346,860)	10

Table 5: Full dataset details. Students with < 20 interactions are removed. In parenthesis, we list the numbers before preprocessing/filtering. R is the repetition number of the training-meta partition split to reduce variance.

6 Datasets and Preprocessing Details

We compare our models on five benchmark public datasets: EdNet, Junyi, Eedi-1, Eedi-2, and ASSISTments; the Eedi-2 dataset has been used in [Wang *et al.*, 2020b]. We remove students having less than 20 interactions; further, in case of duplicate questions, we keep only the first interaction. In Table 5, we list the number of students, the number of questions, and the number of interactions. We also list the number of total students and interactions in each dataset in parentheses (before filtering); we do not need to preprocess Eedi-1 and Eedi-2 datasets, since they contain single interaction for each question and maintain more than 50 total interactions for each student. Following [Wang *et al.*, 2020b], for smaller datasets, for each students in the validation and testing set, we created multiple (R) partitions of training ($\Omega_i^{(1)}$, 80%) and meta set (Γ_i , 20%) to reduce the variance in the final estimate; the number of repetitions (R) for each dataset is added in Table 5. **The (training-meta question set) partitions for the validation and testing students are exactly the same for all models.**

7 Networks and Hyper-parameters

For training IRT-Active and IRT-Random, we use l_2 -norm regularization $\lambda_1 \in \{10^{-3}, 10^{-6}, 10^{-10}\}$. For the IRT-Active, we compute the student ability parameters using the responses and a l_2 -norm regularization $\lambda_2 \in \{10, 1, 10^{-1}, 10^{-2}, 10^{-4}, 0\}$ to penalize deviation from the mean student ability parameter in the training dataset. For the BiNN model, the global response model γ consists of a input vector $\mathbf{w} \in \mathbb{R}^{256}$, and a network with one hidden layer of 256 nodes with weight $\mathbf{W}^1 \in \mathbb{R}^{256 \times 256}$, and a output layer with weight $\mathbf{W}^2 \in \mathbb{R}^{Q \times 256}$ (for simplicity, we are ignoring the biases). The response model computes probability of correctness for question j with student-specific parameter $\theta_i := \{\mathbf{w}_i, \mathbf{W}_i^1, \mathbf{W}_i^2\}$ as, $g(j, \theta_i) = \sigma(\mathbf{W}_i^2 \text{Dropout}(\text{ReLU}(\mathbf{W}_i^1 \mathbf{w}_i)))[j]$, where $\mathbf{x}[j]$ represents the j^{th} dimension of the vector \mathbf{x} , and the sigmoid function $\sigma(\cdot)$ operates element-wise. We keep part of the network ($\mathbf{W}^1, \mathbf{W}^2$) fixed in the inner-level and only adapt the student-specific parameter \mathbf{w} . We use Adam optimization for all question specific parameters in BiIRT and BiMLP models

with the learning rate set to 10^{-3} . For the rest of the parameters of γ (student-specific, that are adapted in the inner-level), we optimize the global values using SGD optimization with the learning rate set to 10^{-4} and momentum set to 0.9. We use a fixed $K = 5$ number of GD steps to optimize the inner-level objective; we find student-specific parameters quickly converge within a few steps with a slightly high learning rate $\alpha \in \{0.05, 0.1, 0.2\}$. The question selection network parameters are optimized using Adam optimizer with learning rate $\eta_2 \in \{0.002, 0.0002\}$. For the unbiased gradient, we take 4 off-policy gradient steps at the end of the selection process, and set the clipping parameter ϵ to 0.2 in all of our experiments, following [Schulman *et al.*, 2017]. We set batch size fixed for all models in each dataset based on the GPU memory and train all the models until the validation performances do not improve for some number of epochs. The required GPU memory is proportional to the number of questions. We fix batch size to 128 for the Eedi-1, Junyi, and ASSISTments datasets, fix batch size to 200 for the EdNet dataset and fix batch size to 512 for the Eedi-2 dataset.

For BiIRT/BiNN-Active methods, we need to take $K = 5$ GD steps in the inference time whereas -Approx and -Unbiased amortizes the cost using the neural network ϕ . Thus, we observe higher inference time for the -Active method compared to -Approx and -Unbiased methods. On average, training the -Approx and -Unbiased model takes 10-20 hours, training the BiIRT/BiNN-Active models take 15-30 hours, and training the BiIRT/BiNN-Random models take 5-10 hours on the larger datasets (Eedi-1, Junyi, and EdNet).

8 Additional Experimental Results

In Table 7, we list 5-fold mean and std accuracy numbers for all models on all datasets. Further, in Table 6, we list 5-fold mean and std AUC numbers for all models on all datasets. We observe similar trends for both of these metric. In Figure 5, we provide our ablation experiment plots using AUC metric.

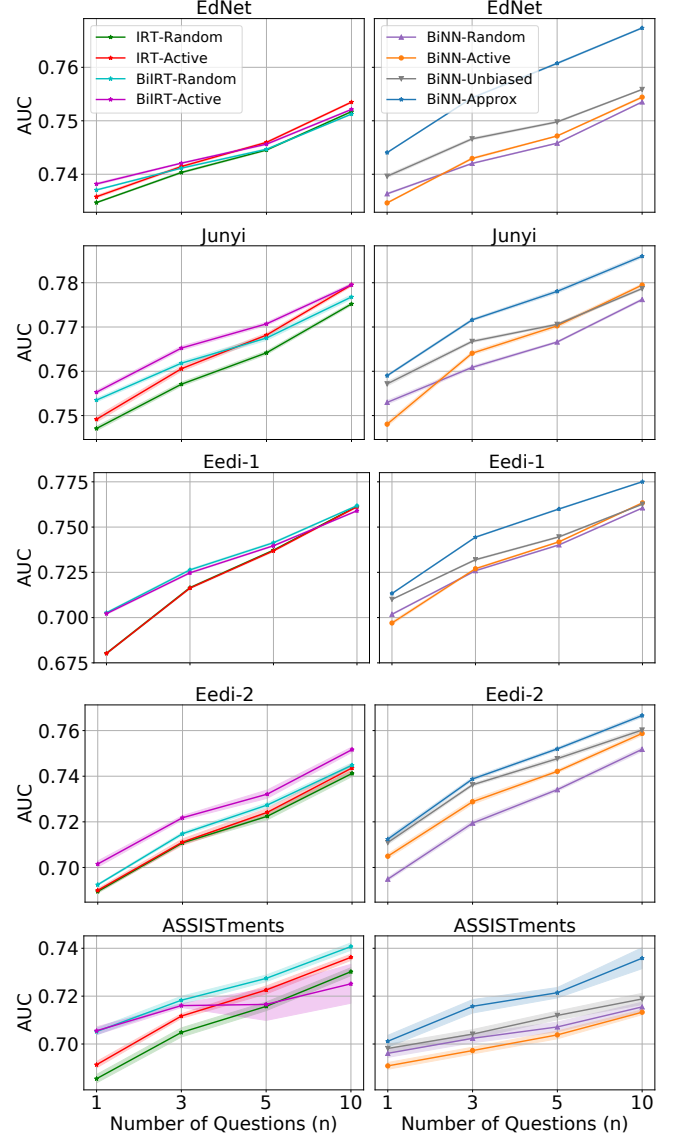


Figure 5: Average AUC (dark lines) and 5-fold standard deviation (light fill lines) on all datasets. First column compares IRT vs BiIRT models; second column compares all BiNN models.

Dataset	Sample	IRT-Random	IRT-Active	BiIRT-Random	BiIRT-Active	BiIRT-Unbiased	BiIRT-Approx	BiNN-Random	BiNN-Active	BiNN-Unbiased	BiNN-Approx
EdNet	1	73.47±0.01	73.58±0.02	73.71±0.01	73.82±0.02	74.14±0.02	74.34±0.01	73.63±0.01	73.46±0.01	73.96±0.03	74.41±0.01
	3	74.03±0.02	74.14±0.02	74.11±0.01	74.21±0.01	74.49±0.01	75.26±0.01	74.2±0.01	74.3±0.02	74.66±0.02	75.43±0.01
	5	74.45±0.01	74.6±0.02	74.47±0.01	74.56±0.02	74.77±0.01	75.68±0.01	74.58±0.01	74.72±0.01	74.98±0.02	76.07±0.01
	10	75.17±0.01	75.35±0.01	75.13±0.01	75.21±0.01	75.39±0.01	76.35±0.01	75.35±0.01	75.44±0.02	75.59±0.01	76.74±0.01
Junyi	1	74.71±0.05	74.92±0.06	75.35±0.04	75.53±0.04	75.67±0.04	75.91±0.03	75.3±0.04	74.81±0.05	75.72±0.05	75.9±0.02
	3	75.71±0.04	76.06±0.06	76.18±0.04	76.52±0.04	76.71±0.04	77.11±0.04	76.09±0.03	76.41±0.04	76.67±0.03	77.16±0.02
	5	76.42±0.04	76.82±0.04	76.75±0.05	77.07±0.04	77.07±0.04	77.69±0.04	76.66±0.02	77.03±0.04	77.06±0.02	77.8±0.04
	10	77.52±0.03	77.95±0.03	77.68±0.05	77.95±0.03	77.86±0.04	78.45±0.03	77.62±0.02	77.95±0.03	77.87±0.02	78.6±0.04
Eedi-1	1	68.03±0.04	68.02±0.04	70.27±0.04	70.22±0.05	70.95±0.02	71.34±0.04	70.18±0.03	69.7±0.08	71.01±0.02	71.33±0.02
	3	71.65±0.04	71.63±0.05	72.64±0.05	72.47±0.04	73.26±0.06	74.21±0.03	72.59±0.03	72.7±0.04	73.2±0.04	74.44±0.03
	5	73.72±0.03	73.69±0.05	74.14±0.05	73.97±0.03	74.54±0.03	75.47±0.03	74.01±0.03	74.18±0.03	74.46±0.03	76.0±0.03
	10	76.14±0.02	76.12±0.04	76.18±0.02	75.9±0.03	76.34±0.02	77.07±0.03	76.06±0.02	76.34±0.02	76.26±0.02	77.51±0.02
Eedi-2	1	68.95±0.07	69.0±0.08	69.24±0.03	70.15±0.13	70.64±0.05	70.81±0.08	69.49±0.07	70.49±0.1	71.09±0.09	71.24±0.09
	3	71.08±0.05	71.11±0.14	71.48±0.07	72.18±0.09	73.11±0.07	73.37±0.08	71.94±0.1	72.88±0.11	73.62±0.06	73.88±0.07
	5	72.24±0.16	72.42±0.17	72.74±0.1	73.21±0.18	74.19±0.06	74.55±0.12	73.41±0.07	74.21±0.07	74.77±0.08	75.2±0.06
	10	74.12±0.14	74.36±0.17	74.48±0.1	75.17±0.12	75.37±0.13	75.96±0.09	75.18±0.09	75.87±0.09	76.02±0.08	76.63±0.08
ASSISTments	1	68.55±0.17	69.14±0.13	70.52±0.13	70.55±0.16	71.0±0.16	71.33±0.24	69.61±0.17	69.09±0.14	69.81±0.18	70.12±0.25
	3	70.48±0.19	71.17±0.1	71.83±0.16	71.6±0.08	72.35±0.19	73.16±0.15	70.24±0.19	69.73±0.14	70.41±0.19	71.57±0.28
	5	71.57±0.18	72.26±0.16	72.74±0.14	71.65±0.67	73.1±0.18	73.71±0.19	70.71±0.2	70.38±0.16	71.19±0.18	72.14±0.22
	10	73.02±0.17	73.62±0.13	74.07±0.15	72.52±0.81	74.38±0.18	74.66±0.22	71.55±0.2	71.33±0.1	71.88±0.23	73.59±0.44

Table 6: 5-fold mean and standard deviation for all models on all datasets using AUC metric.

Dataset	Sample	IRT-Random	IRT-Active	BiIRT-Random	BiIRT-Active	BiIRT-Unbiased	BiIRT-Approx	BiNN-Random	BiNN-Active	BiNN-Unbiased	BiNN-Approx
EdNet	1	69.99±0.01	70.08±0.02	70.87±0.01	70.92±0.01	71.12±0.01	71.22±0.01	70.81±0.01	70.76±0.01	71.01±0.02	71.22±0.01
	3	70.49±0.01	70.63±0.01	71.07±0.01	71.16±0.08	71.35±0.01	71.72±0.01	71.11±0.01	71.23±0.01	71.4±0.02	71.82±0.01
	5	70.84±0.01	71.03±0.01	71.26±0.01	71.37±0.01	71.45±0.01	71.95±0.01	71.32±0.01	71.47±0.01	71.56±0.01	72.17±0.01
	10	71.41±0.01	71.62±0.01	71.63±0.01	71.75±0.01	71.79±0.0	72.33±0.01	71.75±0.01	71.87±0.01	71.89±0.01	72.55±0.01
Junyi	1	74.38±0.06	74.52±0.07	74.8±0.06	74.93±0.05	74.97±0.06	75.11±0.05	74.78±0.05	74.55±0.06	75.02±0.06	75.1±0.05
	3	74.93±0.06	75.19±0.07	75.24±0.06	75.48±0.07	75.53±0.06	75.76±0.06	75.21±0.06	75.4±0.05	75.55±0.04	75.83±0.05
	5	75.33±0.06	75.64±0.06	75.55±0.07	75.79±0.06	75.75±0.05	76.11±0.06	75.52±0.05	75.75±0.06	75.74±0.04	76.19±0.05
	10	75.96±0.06	76.27±0.06	76.06±0.06	76.28±0.05	76.19±0.06	76.49±0.06	76.05±0.05	76.29±0.06	76.18±0.06	76.62±0.06
Eedi-1	1	66.8±0.05	66.92±0.05	68.18±0.06	68.22±0.06	68.61±0.04	68.82±0.06	68.11±0.05	67.83±0.08	68.66±0.04	68.78±0.04
	3	68.63±0.05	68.79±0.06	69.39±0.05	69.45±0.05	69.81±0.05	70.3±0.05	69.33±0.05	69.53±0.05	69.71±0.05	70.45±0.05
	5	69.94±0.04	70.15±0.07	70.21±0.05	70.28±0.05	70.47±0.04	70.93±0.05	70.12±0.04	70.37±0.05	70.39±0.04	71.37±0.05
	10	71.48±0.03	71.72±0.05	71.44±0.04	71.45±0.04	71.57±0.05	72.0±0.05	71.32±0.04	71.7±0.05	71.44±0.03	72.33±0.04
Eedi-2	1	63.65±0.12	63.75±0.12	64.16±0.07	64.83±0.12	65.22±0.04	65.3±0.08	64.31±0.07	65.1±0.09	65.55±0.07	65.65±0.06
	3	65.24±0.11	65.25±0.19	65.84±0.09	66.42±0.09	67.09±0.09	67.23±0.08	66.15±0.1	66.94±0.1	67.55±0.06	67.79±0.05
	5	66.24±0.12	66.41±0.13	66.72±0.1	67.35±0.1	67.91±0.07	68.23±0.09	67.29±0.06	67.98±0.06	68.43±0.07	68.82±0.05
	10	67.71±0.1	68.04±0.12	68.04±0.1	68.99±0.09	68.84±0.1	69.47±0.06	68.68±0.07	69.34±0.08	69.47±0.05	70.04±0.04
ASSISTments	1	65.45±0.2	66.19±0.19	68.63±0.24	68.69±0.27	69.03±0.24	69.17±0.3	67.89±0.23	67.44±0.18	67.98±0.25	68.0±0.25
	3	67.52±0.25	68.75±0.15	69.42±0.24	69.54±0.21	69.78±0.23	70.21±0.19	68.23±0.24	68.13±0.18	68.33±0.22	68.73±0.25
	5	68.55±0.25	69.87±0.21	70.03±0.21	69.79±0.32	70.3±0.26	70.41±0.23	68.52±0.24	68.68±0.19	68.7±0.21	69.03±0.25
	10	69.95±0.2	71.04±0.22	70.88±0.2	70.66±0.35	71.17±0.21	71.14±0.24	68.9±0.21	69.24±0.19	69.04±0.24	69.75±0.3

Table 7: 5-fold mean and standard deviation for all models on all datasets using accuracy metric.