Search-Efficient Computerized Adaptive Testing

Yuting Hong
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
yutingh@mail.ustc.edu.com

Shiwei Tong
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
tongsw@mail.ustc.edu.cn

Wei Huang
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
ustc0411@mail.ustc.edu.cn

Yan Zhuang
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
zykb@mail.ustc.edu.cn

Qi Liu
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
qiliuql@ustc.edu.cn

Enhong Chen*
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
cheneh@ustc.edu.cn

Xin Li
University of Science and Technology of China & Artificial Intelligence Research Institute, iFLYTEK Co., Ltd
leexin@ustc.edu.cn

Yuanjing He
Open University of China
heyuanjing@ouchn.edu.cn

ABSTRACT

Computerized Adaptive Testing (CAT) arises as a promising personalized test mode in online education, targeting at revealing students' latent knowledge state by selecting test items adaptively. The item selection strategy is the core component of CAT, which searches for the best suitable test item based on students' current estimated ability at each test step. However, existing selection strategies behave in a brute-force manner, which results in the time complexity being linear to the number of items (N) in the item pool, i.e., $O(N)$. Thus, in reality, the search latency becomes the bottleneck for CAT with a large-scale item pool. To this end, we propose a Search-Efficient Computerized Adaptive Testing framework (SECAT), which aims at enhancing CAT with an efficient selection strategy. Specifically, SECAT contains two main phases: item pool indexing and item search. In the item pool indexing phase, we apply a student-aware spatial partition method on the item pool to divide the test items into many sub-spaces, considering the adaptability of test items. In the item search phase, we optimize the traditional single-round search strategy with the asymptotic theory and propose a multi-round search strategy that can further improve the time efficiency. Compared with existing strategies, the time complexity of SECAT decreases from $O(N)$ to $O(logN)$. Across two real-world datasets, SECAT achieves over 200x speed up with negligible accuracy degradation.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**; • **Social and professional topics → Computing education**.

KEYWORDS

Computerized Adaptive Testing; Educational Resource Search; Educational Measurement; Cognitive Diagnosis

ACM Reference Format:
Yuting Hong, Shiwei Tong, Wei Huang, Yan Zhuang, Qi Liu, Enhong Chen, Xin Li, and Yuanjing He. 2023. Search-Efficient Computerized Adaptive Testing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3615049

1 INTRODUCTION

With the prevalence of intelligent educational systems, Computerized Adaptive Testing (CAT) has been a crucial issue in many real-world scenarios such as educational measurement, game, and job recruitment [23, 36, 40]. CAT aims to uncover students'/test takers' knowledge state by selecting items adaptively and it has been applied in many standard test organizations, such as Graduate Management Admission Test (GMAT) [34] and Graduate Record Examination (GRE) [31]. Compared with paper-pencil tests, CAT needs fewer items to reach the same measurement accuracy[16].

As shown in Figure 1, CAT consists of three components: (1) An **item pool** is preloaded before the test starts. It contains items with

[*] Corresponding Author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3615049

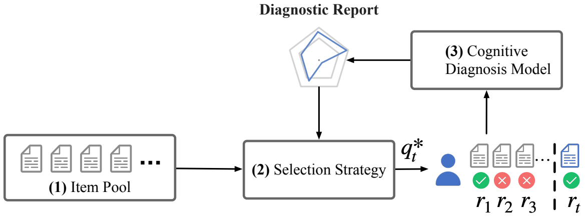

Figure 1: Overview of CAT workflow.

calibrated item parameters which are used for item selection and ability estimation. (2) A **selection strategy** is applied to determine the most suitable test item from the item pool at each test step based on the currently diagnosed state. (3) A **cognitive diagnosis model (CDM)** uncovers the students' knowledge state based on the preceding responses. The workflow of the CAT system is as follows. Given a student, at step t, the selection strategy first selects an item q_t^* based on the current state $\hat{\theta}_t$. Then, the student reads and answers it. After receiving the response r_t, CDM updates the current state to $\hat{\theta}_{t+1}$ according to the responses $\{r_1, r_2, r_3, ..., r_t\}$. The procedure repeats until meeting the termination criteria like reaching the maximum testing length. The final diagnosed ability will be produced as a testing result. As a key component in the testing procedure, the selection strategy has received much effort from researchers. Previous works put stress on how to promote effectiveness by optimizing the score but ignore the latency issue, which results in the search latency becoming an obstacle for a large item pool. Concretely, contemporary selection strategies simply use a score function to calculate the informativeness score of each item, which leads to the time complexity of the exhaustive search being linear to the scale of the item pool, i.e., $O(N)$. For instance, assuming a 100,000-size item pool which is common in online intelligent education systems, selecting the most informative item by Fisher information [26] (a classic informativeness score function) in a brute-force manner costs more than 10 seconds at each step on 2.20 GHz Intel CPU, which is unacceptable in real-world scenarios. Therefore, in practice, well-known organizations like GMAT [34] and GRE [31] require experts to reduce the item candidates by manually developing filtering rules, which is labor-intensive and limits the effectiveness of CAT to a certain degree. To cope with it, promoting the time efficiency of the selection strategy becomes an urgent issue in CAT.

Inspired by successful methods to tackle the efficiency issue of item selection in other areas (e.g., recommendation [39], information retrieval [11, 28]), a potential solution is to divide items into sub-spaces and efficiently search the suitable item from restricted sub-spaces, which is known as Space partitioning [8]. However, it's difficult to design an efficient selection strategy in CAT due to the following challenges. Firstly, in CAT, the item partition should be student-aware. For example, given a specific student, the suitable items for him/her should be divided into the same sub-space. Constructing an index barely on similarity distance ignores the relevance of items to students. Thus, items' adaptability to different students should be considered in the indexing phase of the item pool. Secondly, the score function should be a valid metric such as dot product. However, the information quantity function is invalid in Euclidean Space, which becomes an obstacle to dividing items into sub-spaces. How to address the complex form of the score function becomes a vital problem. Thirdly, item selection is multi-round dependent. To obtain an accurate estimate, CAT selects items step by step over the current estimate. The multi-round selections by recursive estimates in CAT may cause redundant checks with sub-spaces due to similar estimates.

To this end, we propose a Search-Efficient Computerized Adaptive Testing framework (SECAT). Specifically, we first optimize the spatial partition method by considering the adaptability of the item. In this way, the entire selection space ($O(N)$) is recursively partitioned into many sub-spaces of logarithmic size ($O(logN)$), and items with similar selection possibility can be divided into the same sub-spaces. Secondly, to deal with the complex form of the score functions in selection strategies, we formulate a general score function by distilling knowledge from existing strategies. This allows us to divide items into sub-spaces on a valid metric. Thirdly, we utilize the asymptotic theory to reduce the redundant look-ups and further restrict the candidate sub-spaces. When the current estimate is similar to one of the preceding estimates, we select items from the corresponding sub-spaces that generated the best suitable items in the preceding steps. Our approach reduces the linear time complexity of the exhaustive search ($O(N)$) to logarithmic time complexity ($O(logN)$). For example, to select the most informative item from a 100,000-size item pool, we only need to compute the informativeness score of 83 items on average in SECAT at each step, which cost less than 0.1 seconds. Our approach achieves similar accuracy as the brute-force search with more than 200x speed up. To validate the effectiveness and efficiency of SECAT, we conduct experiments on two real-world datasets from educational systems.

2 RELATED WORK

2.1 Computerized Adaptive Testing

CAT is composed of an item pool, a cognitive diagnosis model and a selection strategy. An item pool is preloaded as test content in CAT system, and the number of test items can largely affect the test fairness and quality [14]. During testing, cognitive diagnosis model (CDM) [9, 13] and selection strategy work alternatively until the selection criteria is satisfied. Representative CDMs involve traditional Item Response Theory (IRT) [7] and recent deep learning models (e.g., NCD [41]). Selection strategy is the core component of CAT, which consists of two parts: the score function that measures the informativeness of items and the search strategy that finds the most informative item. Previous works focus on the design of score function and search in a brute-force way. The existing score functions can be divided into two categories: maximum informativeness score functions and data-driven score functions. Maximum informativeness score functions are designed manually to quantify the informativeness. Lord [26] proposed a score function to quantify item informativeness with the Fisher information (FSI). Also, Chang and Ying [4] used Kullback–Leibler (KL) information as the score function in CAT. FSI and KLI are designed on IRT family models. Bi [2] proposed a Model-Agnostic framework designing the score function according to expected model change, which is agnostic with the underlying CDM. Data-driven score functions are learned from data and they are inspired by reinforcement learning [24]. BOBCAT[10] was proposed to directly learn a data-driven score function from training data by recasting CAT as a bilevel optimization problem in the meta learning. NCAT [44] formally redefined CAT as a reinforcement learning problem and directly learns the score function from real-world data. However, data-driven score functions are trained by reinforcement learning, which is far more computationally intensive on large datasets than on small ones. It's difficult to apply them to the large item pool because of too much learning overhead. What's more, the learned score functions are prone to bias in historical data [10]. Therefore, we focus on how to reduce the time complexity of strategies with maximum informativeness score function in this paper.

2.2 Efficient Search

Search efficient methods [17] have been researched to retrieve items accurately and efficiently, which can be organized into three categories. The first category is hash-based methods [27, 37], which project the data from the original space to Hamming space. Some hash based methods like Semantic Hashing [12] learn codes from data. Almost all the hash-based methods suffer from severe information loss, leading to low accuracy of recommendation. The second category is quantization-based methods [18, 20, 30]. Product quantization [19] decomposes the space into sub-spaces, suitable for high dimensional scenarios. Product quantization is usually used for similarity search on Euclidean distance, yet LightRec [25] utilizes product quantization to propose a Memory and Search-Efficient framework based on dot product. The third category is the graph-based method which improves retrieval efficiency through the search of neighboring nodes, such as HNSW[29]. The fourth category is tree-based methods using spatial partition structures, such as KD-Tree [1] and Ball-tree [21, 33, 43]. Koenigstein [22] built a metric tree from item representation, providing exact top-k recommendation based on dot product. SECAT belongs to this taxonomy. Instead of constructing the metric tree barely on item similarity, we consider items' adaptability to different students during spatial partition. In the search phase, we exploit the dependency of multi-round searches in CAT to further improve the time efficiency.

3 PRELIMINARIES

3.1 CAT Components

Specifically, CAT consists of three components: (1) An **item pool** Q consists of items with calibrated parameters, such as difficulty and discrimination. Before testing, items are assigned to professional testers' to collect answers, so that items can be calibrated and a set of user abilities can be obtained as a by-product. The scale of item pool should be considered for three reasons. First of all, a large item pool has been proven to be more reliable because of higher accuracy [14]. An important requirement for adaptive testing is providing sufficient items with various difficulty parameters, so that students of different ability levels can be accurately diagnosed. Second, a large item pool is beneficial to testing fairness and security by controlling the exposure rate. Third, with substantial available items, large item pools are more common in online educational intelligent systems. Since search latency is linear to the increasing scale of item pool in existing selection strategies, reducing the search latency becomes an urgent issue in the large-scale item pool.

(2) A **cognitive diagnosis model** M is used to uncover students' knowledge state based on their responses. The most widely used CDM is Item Response Theory (IRT). IRT uses a unidimensional value θ_i to represent the latent feature of student i and compute the possibility the student i answer the item j correctly:

$$P(r_{i,j} \mid \theta_i, a_j, b_j) = \frac{1}{1 + e^{-1.7a_j(\theta_i - b_j)}}, \tag{1}$$

where $r_{i,j}$ is student i's response to item j (1 indicates a correct response). Each item is represented by two parameters, a_j is the discrimination of the item j and b_j is the difficulty of the item j. Recently, Neural Cognitive Diagnosis Model (NCD) [41] has been proposed, which leverages the deep neural network to model the multidimensional latent trait of the student.

(3) A **selection strategy** is the most important component that selects the next item from the item pool depending on current estimated student ability $\hat{\theta}_t$ at step t. The selection strategy consists of the score function S that measures the informativeness of the item and the search strategy that find the most informative item q_t^*:

$$q_t^* = \arg\max_{q_j \in Q} S\left(\hat{\theta}_t, q_j\right). \tag{2}$$

The search efficiency is difficult to improve due to the complex form of score function. The score function is manually designed and it describes the interaction between the student and the item in a complex way. For example, FSI [26] defined the score function as:

$$S(\hat{\theta}) = \frac{\left[P'(\hat{\theta})\right]^2}{P(\hat{\theta})(1 - P(\hat{\theta}))}, \tag{3}$$

where $P(\hat{\theta})$ denotes the possibility of correct response of the given estimate $\hat{\theta}$ and $P'(\hat{\theta})$ is the derivative of $P(\hat{\theta})$. In conclusion, the forms of score functions in the selection strategies are quite different from the similarity functions (dot product, cosine similarity) and they are invalid in Euclidean space, which becomes an obstacle to the improvement of the time efficiency in the search phase.

3.2 Problem Definition

As mentioned above, CAT system includes an item pool of N-size items $Q = \{q_1, q_2, ..., q_N\}$. Given a new student s_i, at step t, the student's proficiency is estimated based on preceding responses by a cognitive diagnosis model M:

$$P(r_{i,j} = 1 \mid \theta_i, q_j), \tag{4}$$

where θ_i is the ability of the student and $r_{i,j}$ is student i's response to item j (1 indicates a correct answer). We use $M, q_j, r_{i,j}$ to get estimated $\hat{\theta}$. Afterward, a selection strategy selects the item with max informativeness on the current estimated proficiency $\hat{\theta}$ as shown in Equation (2).

Supposing the time complexity of computing the score function $S(\hat{\theta})$ is D, traditional strategies calculate the information of each item in the N-size item pool and select the most informative item in a brute-force way, so the time complexity of selection at each step is $O(ND)$. Our goal is to reduce search latency by narrowing the selection scale N with negligible loss of testing accuracy.

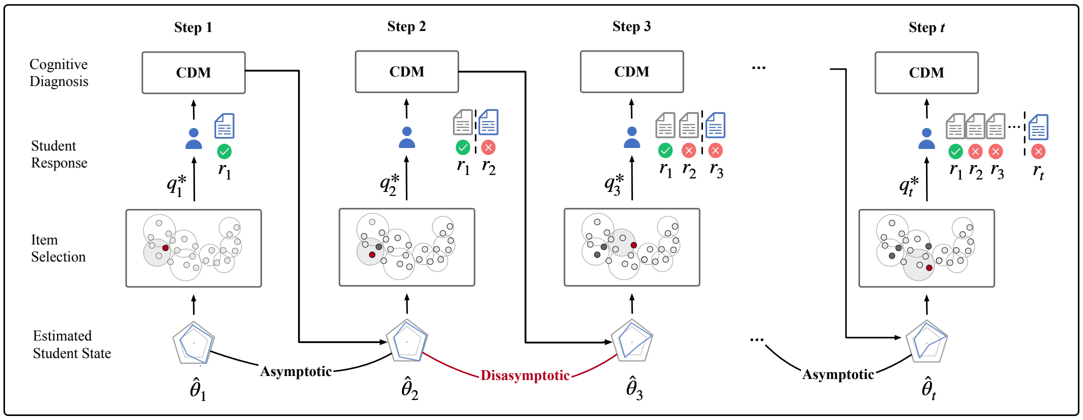

Figure 2: Overview of SECAT workflow.

4 METHODOLOGY

4.1 Overview

SECAT contains two main phases: item pool indexing and item search. In the indexing phase, we build a spatial partition tree on the item pool, which divides the items into binary sub-spaces recursively with the consideration of their adaptability. In the item search phase, we propose an efficient multi-round search strategy to enhance the existing selection strategies. As shown in Figure 2, if the current estimate is asymptotic, we search from the leaves of selected items in preceding rounds, otherwise, we search from the entire tree. Our framework SECAT can be applied to all the score functions of selection strategies, except for NCAT [44] and BOBCAT [10], since they change the paradigm of CAT. In Section 4, we will first introduce how to build an spatial partition tree on the item pool. Then we illustrate the efficient search strategy on the premise of the indexing pool. Finally, we conduct a time complexity analysis of the proposed selection strategy.

4.2 Item Pool Indexing

In the traditional CAT setting, the item pool is a collection of unstructured items. With the goal of reducing search latency, we need to index the item pool with a data structure before testing.

4.2.1 Student-aware Spatial Partition .

We first partition the items into sub-spaces to restrict the selection space. The complex score function of the selection strategy is an important issue in efficient search. We will address the problem in Section 4.2.2 afterward. In this section, we divide the space based on the Euclidean distance. Inspired by previous works [6, 22], we build a metric tree for the item pool by space partition. The root tree node contains the entire items in the item pool and the items are partitioned into binary hyperspheres (balls) by a hyperplane. By recursive partitions, all the items are split into the smallest sub-spaces, denoted as leaves.

In traditional partition method introduced by Andrew Moore [32]. Each node x is split by following steps: 1) Choose two furthest points A, B from each other as two centroids. 2) Define a hyperplane by centroids of two sub-partitions:

$$y(x) = w * x + b, \tag{5}$$

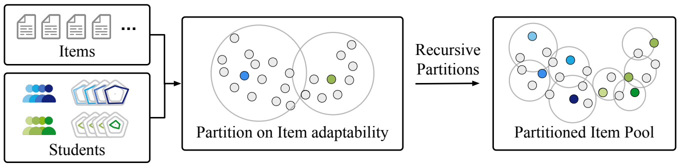

Figure 3: The illustration of student-aware spatial partition.

where,

$$w = B - A,\ b = -\frac{1}{2}(\|B\|^2 - \|A\|^2). \tag{6}$$

3) Separate all the points to two sub-partitions by the hyperplane. However, the hyperplane determined by two random farthest points overlooks the items' relevance to students. The objective of dividing the items into two sub-spaces is to reduce extra look ups in both sub-spaces. For example, as shown in Figure 3, students in blue color are proficient at skills and students in green color are unskilled. The items suitable for skilled and unskilled students should be separately into different sub-spaces. By recursive partitions, items suitable for similar proficiency can be divided into the same sub-spaces.

We propose the student-aware spatial partition method in Algorithm 1. We first obtain a set of student abilities Ω in the calibration stage, when items are assigned to professional testers to collect answers[38]. During each partition, instead of picking the two farthest items, we choose two items suitable for the highest ability and lowest ability in the set as two centroids. Afterward, user abilities in Ω are partitioned into two sets respectively for the next partition. Thus, the hyperplane is determined by the two best-suitable items for different students. As a consequence, we can get a more reasonable hyperplane and the well-organized item pool is constructed by recursive partitions.

4.2.2 Score Function Distillation.

The score functions of existing selection strategies are invalid in Euclidean space. With the spatial partition method, we divided all the items into leaves, but finding the possible candidate leaves can be challenging because of the incompatibility of the score function and the metric tree. Therefore, it is especially urgent to bring up a valid score function that can be used to effectively partition spaces. Considering the form of score function on students and items, we adopt the dot product distance

Algorithm 1 Student-aware Spatial Partition Method

```
Require: Item pool $Q = \{q_1, q_2, ..., q_N\}$, Students $\Omega = \{\theta_0, \theta_1, ..., \theta_M\}$, Score function $S$
function MakeTree(Students $\Omega$, Items $Q$)
    $T.Q \leftarrow Q$
    $T.center \leftarrow mean(Q)$
    $T.radius \leftarrow \max_{q_i \in Q} \|T.center - q_i\|$
    //$N_0$ is the maximum number of items in a leaf.
    if $|Q| \leq N_0$ then
        return $T$
    else
        $(w, b) \leftarrow MakeHyperplane(\Omega, Q)$
        $Q_l \leftarrow \{q_i \in Q : w^\top q_i + b \leq 0\}$
        $Q_r \leftarrow Q - Q_l$
        $\theta_m = median_{\theta_i \in \Omega} \|\theta_i\|$
        $\Omega_l \leftarrow \{\theta \in \Omega : \|\theta\| < \|\theta_m\|\}$
        $\Omega_r \leftarrow \Omega - \Omega_l$
        $T.left \leftarrow MakeTree(\Omega_l, Q_l)$
        $T.right \leftarrow MakeTree\Omega_r, Q_r)$
    end if
    return $T$
end function

function MakeHyperplane(Students $\Omega$, Items $Q$)
    $\theta_l \leftarrow \underset{\theta_i \in \Omega}{min} \|\theta_i\|$;
    $\theta_r \leftarrow \underset{\theta_i \in \Omega}{max} \|\theta_i\|$;
    $A \leftarrow argmax_{q_i \in Q} S(\theta_l, q_i)$;
    $B \leftarrow argmax_{q_i \in Q} S(\theta_r, q_i)$;
    $w \leftarrow B - A$;
    $b \leftarrow -\frac{1}{2}(\|B\|^2 - \|A\|^2)$;
    return $(w, b)$
end function
```

to approximate existing strategies by double-tower network:

$$S(\theta, q) \leftarrow f(\theta)^\top g(q), \tag{7}$$

where f means the user tower encoder, g means the item tower encoder and S represents the informativeness such as Fisher and KL information. We use a double-tower [5, 15] structure to distill knowledge from complex computation for informativeness. Concretely, the student's information (estimated ability and historical responses) is fed into the user tower to obtain student representations, and the item's (calibrated difficulty and discrimination parameters) is fed into the item tower for item representations. Both encoding towers can be an embedding module or the state-of-art module to model side information. And the distillation loss is:

$$Loss = \sum_{j=0}^{N} \|f(\theta)^\top g(q_j) - S(\theta, q_j)\|^2. \tag{8}$$

With the double-tower network, we use the dot product to approximate the informativeness score.

4.3 Item Search

In line with the general score function in Section 4.2.2, we use the dot product function to approximate the score function. In this section, we employ a depth-first branch-and-bound algorithm to search for the most informative item with negligible degradation in performance.

4.3.1 Single-round Search Strategy. In the partitioned item pool in Figure 3, we denote B to be the ball(sub-space) of items, and each ball is centered around q_c within radius r. According to Noam Koenigstein [22], there exists an upper bound for the maximum possible dot product in ball B:

$$\max_{q_i \in B} f(\theta)^\top g(q_j) \leq f(\theta)^\top q_c + r \|f(\theta)\|. \tag{9}$$

With Equation (7) (8) in the score function distillation, we have

$$S(\theta, q_i) \approx f(\theta)^\top g(q_j). \tag{10}$$

Hence,

$$\max_{q_i \in B} S(\theta, q_i) \lesssim f(\theta)^\top q_c + r \|f(\theta)\|. \tag{11}$$

In this way, the information quantity of the items in the ball B has a approximate upper bound: $f(\theta)^\top q_c + r \|f(\theta)\|$. We search for item of the most information quantity in a depth-first manner in Algorithm 2. Before the test, all the items has been transformed by item tower encoder g. During the test, we compute the user representation $f(\theta)$ by user tower encoder f. Following the search algorithm in metric tree[22], if the items in a sub-space has a greater upper bound than the current max information quantity, we traversed the children of the sub-space recursively. Otherwise, the sub-space will be pruned and the computation of the items in it can be waived. In fact, only L leaves with greater upper bound are checked. The order of traversing sub-spaces is determined by the upper bound in the node, aiming to find the possible leaves as soon as possible. Therefore, based on the upper bound of items in the sub-spaces, the single-round search strategy reduce the information quantity computation by restricting items in L leaves.

4.3.2 Multi-round Search Strategy. Though we proposed an efficient single-round search strategy to reduce latency at each step, the dependency among multi-round searches has not been exploited. The CAT system provides the best suitable item from the item pool on the current estimate adaptively, and the t-th item is chosen according to the preceding $t-1$ responses $\{r_1, r_2, ..., r_{t-1}\}$. Accordingly, Chang and Ying [42] has demonstrated that the sequence of recursive estimates $\{\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_{t-1}\}$ is asymptotically consistent with the ground truth θ_0:

$$\lim_{n \to \infty} \Pr \left\{ \left| \hat{\theta}_n - \theta_0 \right| \geq \epsilon \right\} = 0, \tag{12}$$

where ϵ is any arbitrary small positive quantity. The asymptotic theory explains that the estimated ability $\hat{\theta}$ approaches the ground truth gradually during the testing. Thus, it's possible that the current estimate is similar to that of the preceding queries. In this occasion, similar estimate may cause redundant selections.

Therefore, we propose a novel selection strategy for multi-round searches by exploiting the search results in previous rounds. When the current estimated ability $\hat{\theta}$ is quite similar to historical queries, we select the item from the candidate leaves of preceding selections. On the contrary, when $\hat{\theta}$ is dissimilar to all the historical queries, we use Algorithm 2 to select the item from the entire metric tree. Therefore, how to quantify query similarity is a crucial issue in the multi-round search problem.

Algorithm 2 Single-round Search Strategy

```
Require: Estimated student ability $\hat{\theta}$, User tower encoder $f$, Item
    tower encoder $g$, Item Tree Node T, Score function S.
Ensure: The selected item T.M
$T.M \leftarrow None$;
SearchTree($\hat{\theta}, T$);
return $T.M$

function SearchTree(Query $\theta$, Item Tree Node $T$)
    if $T.M < f(\theta)^\top T.center + T.radius\|f(\theta)\|$ then
        if $isLeaf(T)$ then
            $q \leftarrow \arg\max_{g(q_j) \in T.Q} S(\theta, q_i)$;
            if $S(\theta, q) > S(\theta, T.M)$ then
                $T.M \leftarrow q$;
            end if
        else
            $I_l \leftarrow f(\theta)^\top T.left.center + T.left.radius\|f(\theta)\|$;
            $I_r \leftarrow f(\theta)^\top T.right.center + T.right.radius\|f(\theta)\|$;
            if $I_l \leq I_r$ then
                SearchTree($\theta, T.right$);
                SearchTree($\theta, T.left$);
            else
                SearchTree($\theta, T.left$);
                SearchTree($\theta, T.right$);
            end if
        end if
    end if
end function
```

Algorithm 3 Multi-Round Search Strategy

```
Require: Estimated student ability $\hat{\theta}_t$, User tower encoder $f$, Item
    Tree Node $T$, Historical queries $\Theta = \{\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_{t-1}\}$, Historical
    candidates $C$, Threshold $\delta$, Score function S.
Ensure: The selected item T.M
$C_t \leftarrow \emptyset$;
if $\max_{\hat{\theta}_{t'} \in \Theta} Sim(\hat{\theta}_t, \hat{\theta}_{t'}) > \delta$ then
    //With the asymptotic query, the selection space is restricted
    in former selected leaves.
    for each $\hat{\theta}_{t'} \in \Theta$ do
        if $Sim(\hat{\theta}_t, \hat{\theta}_{t'}) > \delta$ then
            $C_t \leftarrow C_t \cup C.\hat{\theta}_{t'}$;
        end if
    end for
    $T.M \leftarrow \arg\max_{q_i \in C_t} S(\theta, q_i)$;
else
    SearchTree($\theta$, $T$);
    //Items in the selected leaf are denoted as Mleaf .
    $C.\hat{\theta}_t \leftarrow T.Mleaf$;
end if
return $T.M$
```

As we mentioned in Section 4.2.2, with the distillation score function, the estimated ability $\hat{\theta}$ is transformed to $f(\hat{\theta})$ by user encoder f and the item q is transformed to $g(q)$ by item encoder g. We use dot product form to approximate the score function S:

$$S \approx f(\hat{\theta})^\top g(q) = \|f(\hat{\theta})\|\|g(q)\|cos\alpha, \tag{13}$$

where α is the angle between the transformed student vector and the transformed item vector. In Equation (13), the length of the query vector doesn't affect the search result, which depends on the angle α and item vector length $\|g(q)\|$. In other words, the small angle between two user vectors implies that two estimated abilities have a similar value. Therefore, we first use the cosine function of queries to quantify the query similarity in Definition 1. And we further define asymptotic query by Definition 2 which is used to search from historical candidate leaves.

Definition 1. ***Query Similarity****: Given a student i with the ability θ_i and student j with the ability θ_j, the similarity of students' ability is :*

$$Sim(\theta_i, \theta_j) = cos(f(\theta_i), f(\theta_j)). \tag{14}$$

Definition 2. ***Asymptotic Query****: Given a student i with estimated ability $\hat{\theta}_t$ at step t, similarity threshold δ and the student's historical estimated ability set $\Theta = \{\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_{t-1}\}$, the current selection has an asymptotic query if*

$$\min_{\hat{\theta}_{t'} \in \Theta} Sim(\hat{\theta}_t, \hat{\theta}_{t'}) > \delta. \tag{15}$$

As shown in Algorithm 3, we use a similarity threshold δ to decide whether to search from previous candidate leaves or the entire metric tree. First, the estimated student's proficiency will be used to compare with historical estimates. If the maximum similarity is greater than the threshold δ, which means the selection has an asymptotic query, we search the historical candidate leaves instead of using the single-round search strategy. With efficient multi-round search strategy, redundant searches from the entire item pool can be further reduced.

4.4 Time Complexity Analysis

Supposing that the time complexity of score function $S(\hat{\theta}, q)$ is D and the size of the item pool is N, the time complexity of the exhaustive search at each step is $O(ND)$. In SECAT, we constructed a metric tree on the item pool. Since the metric tree divides N items into binary sub-spaces recursively, the smallest sub-spaces (leaves) have items of $O(logN)$ size. By pruning the sub-spaces with a small upper bound, we assume L leaves are traversed during selection. Therefore, the time complexity is reduced to $O(DLlogN)$ in the metric tree with single-round search. With the multi-round search strategy, the selection space is restricted to selected leaves in previous rounds when the current estimate changes little compared with the preceding estimates. So L is further reduced in the multi-round search strategy, which will be examined by experiments in Section 5.5.

4.5 Relation to ANN methods

Since we use the dot product to approximate the informativeness quantity score. Technically, the approximate nearest neighbor (ANN) search for the max dot product can also be applied in CAT. ANN methods aim to search the approximate items from millions of items, which has outstanding performance in other tasks (CV[35]

Dataset	Eedi	Exam
#Students	118,971	1,897,707
#Items	27,613	122,950
#Response logs	15,867,850	89,106,879
#Response logs per student	133	47
#Response logs per item	575	725

Table 1: Statistics of the datasets

and RS[25]). However, CAT has a higher demand for accuracy since it uses fewer questions than paper-pencil testing. SECAT can reduce the accuracy degradation in two aspects: 1) The backbone of SECAT is Ball Tree, which searches for the exact item for the query. 2) During searching in the leaves, SECAT utilizes the approximate upper bound to compute the information quantity by the vanilla score function S (FSI, KL, EMC). The approximate upper bound is only valid in Ball Tree, so it cannot be applied in other ANN methods. Therefore, ANN could cause extra degradation in accuracy. In conclusion, SECAT can guarantees the accuracy of testing in item searches better.

5 EXPERIMENTS

5.1 Experimental Settings

Datasets. We use two real-world datasets in this experiment, namely Eedi and EXAM. Eedi[1] refers to the dataset in the NeurIPS 2020 Education Challenge. The EXAM dataset was supplied by iFLYTEK Co., Ltd., which contains mathematical exercises and logs of high school examinations. We choose datasets with large item pools to evaluate our proposed method. Table 1 shows the complete statistics of the datasets.

Data Partition. We split student-item interactions into historical data and testing data for different targets. The historical data is collected before testing to calibrate item parameters, such as difficulty and discrimination. We filter out learners with less than 30 response logs for Eedi and Exam respectively in historical data to guarantee the quality of calibrated items. The test data is used to simulate the adaptive testing process in the experiment.

Evaluation Tasks. Following previous works[2, 44], we perform experiments on a simulation study and a student performance prediction task.

1) **Simulation Study**: Traditional CAT studies have a evaluation process called simulation study, which first initializes student abilities and then generates records on the entire item pool with a CDM. During testing, CAT estimates the student ability by adaptively selecting the records. Since CAT selects the item from the entire item pool in simulation study, the search latency is close to real world.
2) **Student Performance Prediction**: We conduct an experiment on the student performance prediction task to examine the performance of CAT. For evaluation, we limit our selection to those items whose response has been recorded in the testing data.

Evaluation Metrics. We use the following two categories of metrics to evaluate the experimental results.

1) **Efficiency Metrics**: Intuitively, we use the time (seconds) spent on selection to measure search time efficiency. To further analyze the time complexity, we use the number of traversed leaves L in the metric tree to estimate the computation cost at each step.
2) **Effectiveness Metrics**: In simulation study, we calculate the mean squared error (MSE) between the estimated parameters and the simulated parameters. In the student performance task, we predict the students' performance on items whose responses have been recorded. From the binary classification perspective, we use accuracy (ACC) and Area Under ROC (AUC) [3] to evaluate different selection strategies.

Baseline Methods. The score function of the selection strategy in CAT relies on Cognitive Diagnosis Model (CDM) as mentioned above. We use Item Response Theory(IRT) [7] and a deep learning-based model(NCD) [41] as the underlying CDM. We use the following state-of-art score functions of selection strategies as baselines.

- **Random**: The random search method is a benchmark to show other methods' improvement.
- **FSI** [26]: Fisher information is the most popular score function designed for IRT.
- **KLI** [4]: KL information is designed to improve accuracy at the beginning of the test. It utilizes Kullback-Leibler information to measure the divergence between two consecutive posteriors of proficiency. It's designed for IRT-based Models.
- **MAAT** [2]: MAAT is inspired by active learning methods, using expected model change (EMC) to quantify the informativeness. It's agnostic to underlying CDMs.

We use **X + SECAT** to denote SECAT with additional student-ware hyperplane and multi-round selection strategy.

Our framework SECAT can be applied to all the score functions, except for data-driven algorithms(NCAT and BOBCAT), since they change the paradigm of CAT and it's difficult to apply them to large item pools. For example, in the dataset Exam, training the selection algorithm with BOBCAT in an epoch takes more than 18 hours on GPU and the entire training requires tens to hundreds of epochs. Also, the turnaround time grows rapidly as the item pool size increases. Therefore, data-driven algorithms are not suitable for the large item pool.

Implementation Details. The threshold for the maximum number of items in a leaf is 50 in Eedi and 100 in Exam for that Exam has more items. The dimension of the student vector and item vector is 15 in the general score function to reduce the inner product computation cost. For the multi-round search strategy, the threshold of query similarity is 0.95 and the related analysis is shown in Section 5.6. All the methods are developed and trained on two 2.20 GHz Intel Xeon E5-2650 v4 CPUs and a TITAN Xp GPU[2].

5.2 Simulation Study

The ultimate goal of CAT is to get the estimate of the student's ability θ. Since the ground truth θ_0 is unknown, we conduct the simulation experiment of proficiency estimation. Specifically, we initialize students' abilities artificially and generate responses on the entire item pool. Since the simulation study is only suitable for those CDMs with simple and explainable parameters, we only conduct the simulation study with IRT.

[1] https://eedi.com/projects/neurips-education-challenge

[2] Our code is available at https://github.com/bigdata-ustc/SECAT.

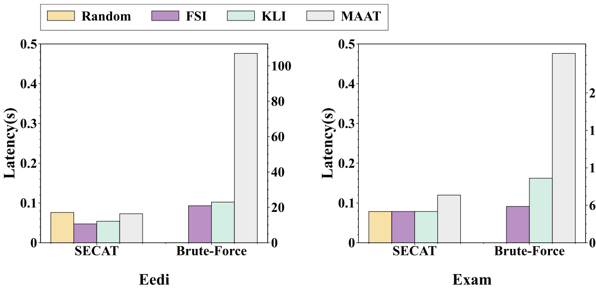

Figure 4: The search latency of simulation study

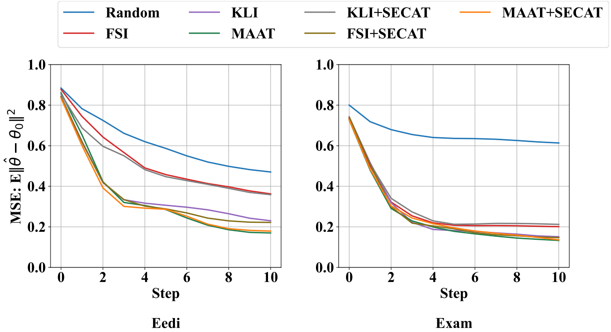

Figure 5: The MSE of simulation study.

5.2.1 Efficiency Analysis. Figure 4 shows the average search latency of brute-force methods and SECAT at one step. We use double Y axes to measure the latency in both datasets because brute-force search methods cost much more time than efficient search methods. Specifically, our proposed SECAT refers to the left axis and the brute-force search method refers to the right axis. Firstly, we can see that SECAT outperforms the brute-force method on both datasets. In the mildest condition, the accumulated latency of brute-force search by FSI score function costs more than 20 seconds, while the accumulated latency of SECAT costs less than 0.1 seconds. Therefore, SECAT achieves over 200x speed up on brute-force method in simulation study. Secondly, although brute-force method on different score functions cost various search latency, SECAT significantly reduces the search latency of all score functions to the order of magnitude of the random search method. For brute-force method, the computation of information quantity in MAAT is more time-consuming than FSI and KLI. For SECAT, it reduces the computation of information quantity by restricting the number of items. By search latency reduction, SECAT makes it possible to select the item from a tremendous pool of items.

5.2.2 Effectiveness Analysis. As shown in Figure 5, we denote the efficient search methods on different score functions as X+SECAT. Firstly, we can see that the X+SECAT can approximate the selection of brute-force methods in MSE metric, which means efficient strategies maintain precision during testing. Secondly, the effectiveness of Random selection in the dataset Exam is inferior to Eedi, while the designed score function (MFI, KLI, MAAT) achieves better performance than Eedi. The reason can be that Exam provides a larger item pool, which covers more potential possible abilities. In conclusion, SECAT plays a crucial role in a large item pool because the relative improvement of the designed score function is more obvious in a larger item pool.

Dataset		Eedi				Exam			
CDM		IRT			NCD	IRT			NCD
Score function		FSI	KLI	MAAT	MAAT	FSI	KLI	MAAT	MAAT
Accumulated latency(s)	Random	0.417	0.417	0.417	0.462	0.758	0.758	0.758	0.823
	Brute-force	1.583	2.742	7.192	18.085	1.375	1.742	3.34	14.952
	SECAT	0.676	0.688	0.704	0.88	0.68	0.69	0.717	0.769
ACC(%)	Random	68.89	68.89	68.89	70.18	76.1	76.1	76.1	76.41
	Brute-force	69.59	69.64	70.35	72.55	78.27	78.05	79.0	80.89
	SECAT	69.75	69.29	70.23	72.38	78.23	78.22	78.97	80.37
AUC(%)	Random	70.93	70.93	70.93	71.27	85.1	85.1	85.1	85.52
	Brute-force	72.07	72.05	72.53	73.11	86.03	86.06	87.38	87.61
	SECAT	71.94	72.14	72.52	72.92	86.0	86.0	87.33	87.56

Table 2: The results of student performance prediction.

Metric	ACC	AUC	MSE
Brute-Force	72.55	73.11	0.17
SECAT	**72.38**	**72.92**	**0.178**
ANNOY	71.94	72.21	0.695
HNSW	72.03	72.67	0.313

Table 3: Comparison with ANN methods on eedi dataset using NCD at step 10.

5.3 Student Performance Prediction

We also conduct an experiment on the student performance prediction task to verify the efficiency and effectiveness of selection strategies. Table 2 reports the accumulated search latency, ACC and AUC at step 10 during the testing. First, SECAT can significantly reduce search latency than the brute-force search method on two datasets. Especially, the brute-force search method relying on NCD-MAAT costs almost 7 seconds in Eedi and 14 seconds in Exam during adaptive testing. In both datasets, SECAT reduces the search latency of MAAT to the order of magnitude of the random search method. Secondly, as shown in Table 2, the negative influence of our proposed SECAT on ACC/AUC is negligible. SECAT achieves competitive performance on prediction performance. For example, SECAT based on NCD-MAAT achieves nearly the same performance as the brute-force method. It means restricted selection item space in SECAT preserves the possible items properly. Moreover, we note that the search latency of the brute-force method on FSI and KLI didn't show a significant difference between SECAT. We believe that this is mainly due to the following reason: In terms of accumulated latency, CAT selects items from the items whose responses have been recorded in testing data, which is a small fraction of the item pool. Therefore, the limited scale of selection space leads to similar accumulated latency.

5.4 SECAT vs ANN Methods

As mentioned in Section 4.5, ANN can also be applied in CAT after the score function distillation. Thus, we compare the effectiveness of

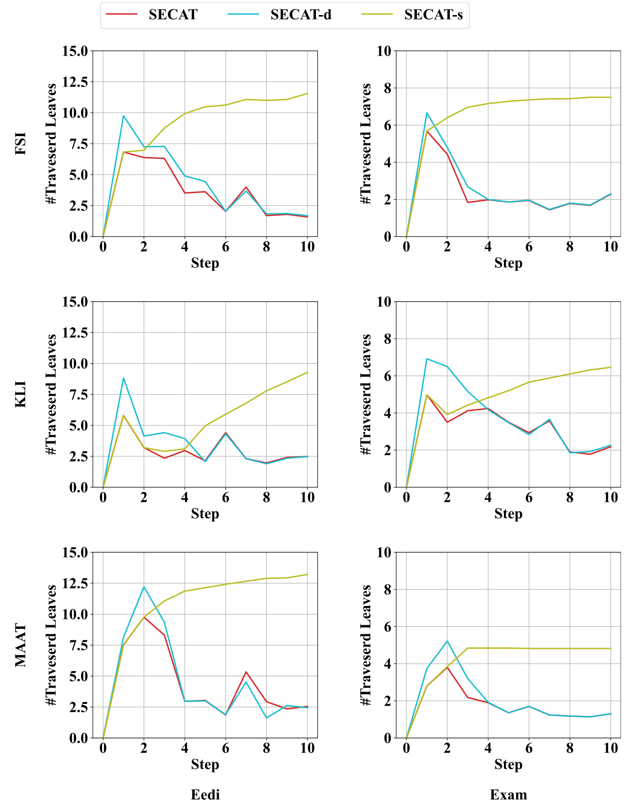

Figure 6: The averaged traversed leaves number at each step.

SECAT with two powerful ANN methods: ANNOY[3] and HNSW[29] in Table 3. We can see that the degradation of SECAT is negligible in all effectiveness metrics while ANNOY and HNSW cause extra degradation in ACC and AUC. Especially in MSE, the estimation of ANN methods exists significant deviation. Since the goal of CAT is accurate parameter estimation, ANN methods are not suitable for CAT. The reason may be that ANN methods are designed to solve for high dimensionality items search at the sacrifice of accuracy. In short, compared to ANN methods, SECAT maintains the testing accuracy and is more suitable for testing scenarios.

5.5 Ablation Study

By performing an ablation study on a simulation study with IRT, we analyze the effectiveness of student-aware spatial partition and multi-round search strategy by comparing the number of traversed leaves. The results are shown in Figure 6. We denote the SECAT without the multi-round search strategy as SECAT-m and SECAT without the student-aware spatial partition method as SECAT-s. Firstly, for SECAT-s, the number of traversed leaves increases as the step increases, which means more leaves need to be explored in the latter steps. The reason is that CAT doesn't allow duplicate selected items in different steps and it explores more possible leaves in the tree in the latter rounds. Secondly, the number of traversed leaves of SECAT decreases in the latter rounds, which means the multi-round selection strategy with the asymptotic query plays a more crucial role in reducing the selection space at the following steps. Thirdly, we can see that SECAT-s searches more leaves for the most suitable item, which means the student-aware hyperplane restricts the selection space at the beginning of testing.

We denote the traversed leaves number as L, and L is decreased to a constant value in SECAT. In conclusion, the time complexity of SECAT-m is $O(DLlogN)$ with L traversed leaves. SECAT reduces the averaged L nearly to a constant value in most rounds. Therefore, the time complexity for SECAT is approximately $O(DlogN)$.

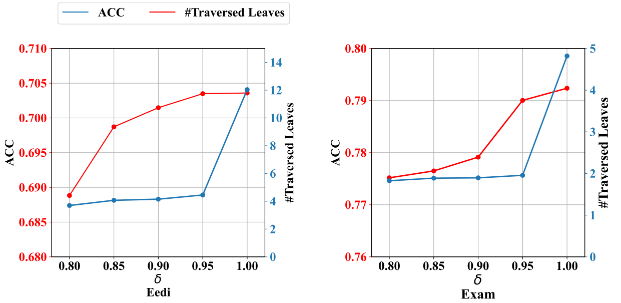

Figure 7: Influence of δ balances the prediction accuracy and search latency.

5.6 Model Parameter Analysis

In SECAT, the trade-off parameter δ in the multi-round search strategy plays an important role in balancing the effectiveness and efficiency by deciding whether to select from the previous leaves or the entire item pool. We carry out the parameter-sensitive experiment with the NCD-MAAT score function on student performance prediction task to see the influence of δ. The δ ranges from 0 to 1. When δ is smaller, the possibility of selecting from a leaf is higher. Conversely, as δ is larger, SECAT tends to select from the entire space. When δ equals 1, the model selects the item with the single-round search strategy.

As shown in Figure 7, when δ increases, the ACC increases. This indicates that properly exploring the items in unselected leaves is beneficial for effectiveness. When δ is too large, the number of traversed averaged leaves glows rapidly, which leads to long search latency. These results show that it's vital to choose a proper threshold δ to balance effectiveness and efficiency.

6 CONCLUSION

In this paper, we proposed a novel Search-Efficient Computerized Adaptive Testing framework for intelligent education systems. To find the possible candidates as fast as possible, we proposed a student-aware space partition method by considering the adaptability of items. Furthermore, we used the asymptotic theory in CAT to utilize preceding responses to help with the next selection. As a general work, SECAT can be applied to many score functions. Extensive experiments have demonstrated the efficiency and effectiveness of SECAT. In future work, we will explore more realistic constraints in CAT such as exposure control.

Acknowledgements. This research was partially supported by grants from the National Key Research and Development Program of China (Grant No. 2021YFF0901003) and the National Natural Science Foundation of China (Grant No. U20A20229).

[3] https://github.com/spotify/annoy

REFERENCES

[1] Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM* 18, 9 (1975), 509–517.

[2] Haoyang Bi, Haiping Ma, Zhenya Huang, Yu Yin, Qi Liu, Enhong Chen, Yu Su, and Shijin Wang. 2020. Quality meets diversity: A model-agnostic framework for computerized adaptive testing. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 42–51.

[3] Andrew P Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition* 30, 7 (1997), 1145–1159.

[4] Hua-Hua Chang and Zhiliang Ying. 1996. A global information approach to computerized adaptive testing. *Applied Psychological Measurement* 20, 3 (1996), 213–229.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[6] Mohamad Dolatshah, Ali Hadian, and Behrouz Minaei-Bidgoli. 2015. Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces. *arXiv preprint arXiv:1511.00628* (2015).

[7] Susan E Embretson and Steven P Reise. 2013. *Item response theory*. Psychology Press.

[8] Xuhui Fan, Bin Li, and Scott Sisson. 2018. The binary space partitioning-tree process. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1859–1867.

[9] Weibo Gao, Qi Liu, Zhenya Huang, Yu Yin, Haoyang Bi, Mu-Chun Wang, Jianhui Ma, Shijin Wang, and Yu Su. 2021. Rcd: Relation map driven cognitive diagnosis for intelligent education systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 501–510.

[10] Aritra Ghosh and Andrew Lan. 2021. Bobcat: Bilevel optimization-based computerized adaptive testing. *arXiv preprint arXiv:2108.07386* (2021).

[11] Artem Grotov and Maarten De Rijke. 2016. Online learning to rank for information retrieval: Sigir 2016 tutorial. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 1215–1218.

[12] Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2019. Unsupervised neural generative semantic hashing. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 735–744.

[13] Johannes Hartig and Jana Höhler. 2009. Multidimensional IRT models for the assessment of competencies. *Studies in Educational Evaluation* 35, 2-3 (2009), 57–63.

[14] Hung-Yu Huang. 2018. Effects of item calibration errors on computerized adaptive testing under cognitive diagnosis models. *Journal of Classification* 35, 3 (2018), 437–465.

[15] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2333–2338.

[16] Yueh-Min Huang, Yen-Ting Lin, and Shu-Chen Cheng. 2009. An adaptive testing system for supporting versatile educational assessment. *Computers & Education* 52, 1 (2009), 53–67.

[17] Won-Seok Hwang, Ho-Jong Lee, Sang-Wook Kim, Youngjoon Won, and Min-soo Lee. 2016. Efficient recommendation methods using category experts for a large dataset. *Information Fusion* 28 (2016), 75–82.

[18] Young Kyun Jang and Nam Ik Cho. 2020. Generalized product quantization network for semi-supervised image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3420–3429.

[19] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 33, 1 (2010), 117–128.

[20] Yannis Kalantidis and Yannis Avrithis. 2014. Locally optimized product quantization for approximate nearest neighbor search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2321–2328.

[21] Seongju Kang, Chaeeun Jeong, and Kwangsue Chung. 2020. Tree-based real-time advertisement recommendation system in online broadcasting. *IEEE Access* 8 (2020), 192693–192702.

[22] Noam Koenigstein, Parikshit Ram, and Yuval Shavitt. 2012. Efficient retrieval of recommendations in a matrix factorization framework. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. 535–544.

[23] Jiatong Li, Fei Wang, Qi Liu, Mengxiao Zhu, Wei Huang, Zhenya Huang, Enhong Chen, Yu Su, and Shijin Wang. 2022. HierCDF: A Bayesian Network-based Hierarchical Cognitive Diagnosis Framework. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 904–913.

[24] Yuxi Li. 2017. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274* (2017).

[25] Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. 2020. Lightrec: A memory and search-efficient recommender system. In *Proceedings of The Web Conference 2020*. 695–705.

[26] Frederic M Lord. 2012. *Applications of item response theory to practical testing problems*. Routledge.

[27] Xu Lu, Lei Zhu, Zhiyong Cheng, Liqiang Nie, and Huaxiang Zhang. 2019. Online multi-modal hashing with dynamic query-adaption. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 715–724.

[28] Denghao Ma, Yueguo Chen, Xiaoyong Du, and Yuanzhe Hao. 2018. Interpreting fine-grained categories from natural language queries of entity search. In *Database Systems for Advanced Applications: 23rd International Conference, DASFAA 2018, Gold Coast, QLD, Australia, May 21-24, 2018, Proceedings, Part I 23*. Springer, 861–877.

[29] Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* 42, 4 (2018), 824–836.

[30] Yusuke Matsui, Yusuke Uchida, Hervé Jégou, and Shin'ichi Satoh. 2018. A survey of product quantization. *ITE Transactions on Media Technology and Applications* 6, 1 (2018), 2–10.

[31] Craig N Mills and Manfred Steffen. 2000. The GRE computer adaptive test: Operational issues. In *Computerized adaptive testing: Theory and practice*. Springer, 75–99.

[32] Andrew Moore. 2013. The Anchors Hierachy: Using the triangle inequality to survive high dimensional data. *arXiv preprint arXiv:1301.3877* (2013).

[33] Parikshit Ram and Alexander G Gray. 2012. Maximum inner-product search using cone trees. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 931–939.

[34] Lawrence M Rudner. 2009. Implementing the graduate management admission test computerized adaptive test. In *Elements of adaptive testing*. Springer, 151–165.

[35] Andrey V Savchenko. 2017. Maximum-likelihood approximate nearest neighbor method in real-time image recognition. *Pattern Recognition* 61 (2017), 459–469.

[36] Shuanghong Shen, Enhong Chen, Qi Liu, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. 2022. Monitoring Student Progress for Learning Process-Consistent Knowledge Tracing. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[37] Benno Stein. 2007. Principles of hash-based text retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 527–534.

[38] Nathan A Thompson and David A Weiss. 2011. A framework for the development of computerized adaptive tests. *Practical Assessment, Research, and Evaluation* 16, 1 (2011), 1.

[39] Manos Tsagkias, Tracy Holloway King, Surya Kallumadi, Vanessa Murdock, and Maarten de Rijke. 2021. Challenges and research opportunities in ecommerce search and recommendations. In *ACM SIGIR Forum*, Vol. 54. ACM New York, NY, USA, 1–23.

[40] Wim J Van der Linden and Peter J Pashley. 2009. Item selection and ability estimation in adaptive testing. In *Elements of adaptive testing*. Springer, 3–30.

[41] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. 2020. Neural cognitive diagnosis for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6153–6161.

[42] Zhiliang Ying and CF Jeff Wu. 1997. An asymptotic theory of sequential designs based on maximum likelihood recursions. *Statistica Sinica* 7, 1 (1997), 75–91.

[43] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1079–1088.

[44] Yan Zhuang, Qi Liu, Zhenya Huang, Zhi Li, Shuanghong Shen, and Haiping Ma. 2022. Fully Adaptive Framework: Neural Computerized Adaptive Testing for Online Education. (2022).