Active Learning with a Budget to Rank Candidates Rated by Disjoint Assessors

Tushar Phule
tusharphule82@gmail.com
Indian Institute of Technology, Madras
India

Pragalbh Vashishtha
pragalbhv@gmail.com
Indian Institute of Technology, Madras
India

Arun Rajkumar
arunr@cse.iitm.ac.in
Indian Institute of Technology, Madras
India

ABSTRACT

We consider the problem of learning to rank a set of candidates which are rated by a set of assessors. In several practical settings (candidate-assessor, for instance), the assessors are divided into groups/panels, and each panel assesses a set of candidates. Assessors might look for different qualities/features in a candidate and base their ratings on their features of interest hence in order to obtain a final ranking, it is important to understand the ratings given by each assessor to every candidate. However, it may not always be feasible for each assessor to evaluate every candidate. In this setup, standard matrix completion algorithms fail to recover the entire matrix meaningfully, so the ranking obtained from such a complete matrix is inaccurate. We consider the case where a small extra budget is available, which an algorithm can actively use to choose a set of (candidate, assessor) pairs to query with the goal of obtaining good rankings of the candidates. We propose two novel algorithms, 1. Query by Candidate Probability and Local Coherence Probability (*OPLP-Query*) and 2. Query by Base Factor and Local Coherence Probability (*BFLP-Query*). These algorithms learn good rankings in an active query-based model and are inspired by two natural but different human assessor models. Specifically, they decide to query a candidate-assessor pair by first choosing a candidate using a certain probabilistic scheme and then choosing the best assessor to query for the chosen candidate based on a local coherence-based probabilistic scheme. We conduct extensive experiments on synthetic and real-world datasets to test our algorithms against several baselines, which show that the proposed algorithms clearly outperform existing baselines for this problem.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Active learning settings**;

KEYWORDS

Ranking, Learning and adaptation, Active Learning

KDD OARS 2023, August 6-10, 2023, Long Beach, California - USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00

ACM Reference Format:
Tushar Phule, Pragalbh Vashishtha, and Arun Rajkumar. 2023. Active Learning with a Budget to Rank Candidates Rated by Disjoint Assessors. In *Proceedings of The 3rd International Workshop on ONLINE AND ADAPTIVE RECOMMENDER SYSTEMS (KDD OARS 2023).* ACM, New York, NY, USA, 10 pages.

1 INTRODUCTION

We consider the problem of learning to rank a set of candidates which are assessed/rated by a set of human assessors. The setup we are interested in is when the assessors are partitioned into disjoint sets, and each set assesses only a subset of candidates. Furthermore, each candidate is assessed by exactly one assessor set. The scenario described is relevant to several real-world applications, such as the admission process for graduate programs at universities or the recruitment of employees by companies, where a limited number of vacancies must be filled from a pool of candidates. A common method for determining the ranking of these candidates is to calculate each candidate's score as the average of their ratings from the panel of assessors they interviewed with and then rank all the candidates based on these scores. If all assessors are *similar* and *homogeneous*, such an approach makes sense and might lead to good rankings. However, in actual assessments, each assessor may have unique criteria for evaluating a candidate, leading to different ratings based on their individual preferences and perspectives. In this case, the scheme described earlier might not be the best way to obtain scores. Even if the assessors were explicitly asked to rate the candidates based on the same set of features, there may be *generous* assessors who tend to rate with larger scores and *strict* assessors who might rate with smaller scores [16, 17]. These variations make the problem of combining the ratings across panels ineffective, and one needs alternate approaches.

In this work, we consider the above-described setup where in addition to the ratings given by the assessor groups to subsets of candidates, we also have a limited extra *budget* which can be used to query new (candidate-assessor) pair (see Figure 1). In the candidate-assessor example, the budget might correspond to requesting ratings from assessors outside the original evaluation panel for a candidate. In practice, one can assume that the videos of the interviews are recorded, and the assessors might be requested to view these videos and give their ratings. However, it is impractical to ask all interviewers to view videos of all interviews they did not attend, and hence one has a limited budget to work with. Thus, the problem we consider is the following: *Given a set of candidates assessed by disjoint assessor sets and a query budget, develop algorithms to adaptively query unrated (candidate-assessor) pairs for ratings to*

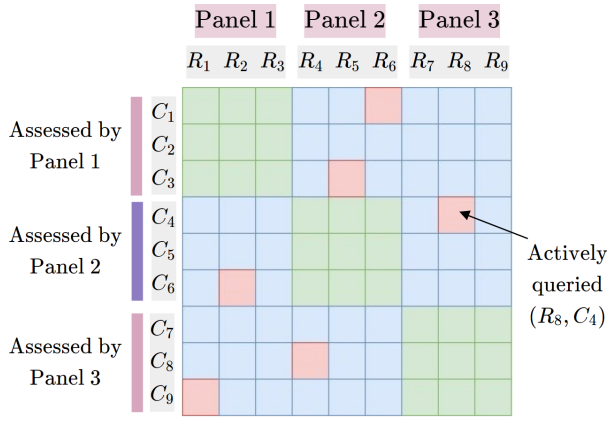


Figure 1: Block Diagonal Structure of candidate-assessor data. Green Entry: candidates assessed by a respective reviewer, Blue: Not evaluated by respective reviewer, Pink: Actively queried entries using available budget. Known entries(green) are spread across the block diagonal as a disjoint set of candidates assessed by a disjoint set of assessors.

obtain good rankings over the candidates, especially focusing on the top k ranks. Figure 1 describes our setup.

Remark: Our objective is to achieve good rankings at the top, as in many practical scenarios (graduate program admits, for instance) only a limited number of positions need to be filled after evaluating multiple applicants.

Our Contributions: We consider two natural assessor models where the underlying true candidate-assessor rating matrix is of low rank. In the first model, the assessor-weightage for the latent features is assumed to be Gaussian distributed, and in a second model inspired by the work of [11] where there is a small set of candidates which act as *base factors* in the sense that the convex hull of their features contains the features of all the candidates. We develop two novel algorithms, which we call, 1. Query by *Candidate Probability and Local Coherence Probability* (OPLP) and 2. Query by, *Base Factor and Local Coherence Probability (BFLP)*. Both these algorithms obtain good rankings by maintaining distributions over candidates then sampling a candidate first and, given the candidate, sampling an assessor according to a different distribution. We conduct extensive experiments on synthetic and real-world datasets (The MIT Interview Dataset) and observe that the performance of the proposed algorithm is significantly better than a variety of baselines.

2 RELATED WORK

We start by considering previous works that are most similar to ours in terms of the setup. [17] considers a setup where the data is in block-structured form like our setup (see Figure 1), and the goal is to obtain a ranking that strictly uniformly dominates the random policy. However, our setup differs from theirs because we assume a statistical assessor model, and our goal is to obtain a ranking that is as close as possible to the true ranking.

[12] work in a setup where a set of assessors rates a set of candidates. In addition to the rating, they also require a confidence score associated with each rating. They assume that the candidate-assessor graph is well connected and infer rankings based on propagating the ratings over the graph. However, our setup translates to a disconnected graph. While the extra queries in our setting allow us to make the graph connected, the goal is not to make it as connected as possible but to make it connected in such a way that the rankings are good w.r.t the underlying assessor model.

[6] proposes the idea of human-in the loop idea for matrix completion, where the human annotation is done with matrix completion. The goal is to appropriately identify a subset of missing entries for manual annotation and aims to lead to a better reconstruction of the incomplete matrix with minimal human effort. The aim is to actively complete a matrix. The main difference between their setup and ours is that our goal is not to complete the matrix but to get good estimates of row sums of the matrix, especially focusing on the top few ranks. An entry of the matrix that may be critical for completing the matrix (where the completion is measured w.r.t the true matrix in some norm) may not necessarily be critical for ranking as the other entries in the corresponding row might be irrelevant.

We next discuss a few related works in matrix completion which are relevant to our work. We describe them in more detail in the preliminary section as needed.

[3–5, 9, 10, 15] solves the problem of recovering a low rank matrix from partially known entries. One popular approach [4, 15] solves the problem of recovering a low rank $n \times n$ matrix from y uniformly sampled entries by solving a convex optimization problem. For a matrix with a rank that is not too large, if the number of sampled entries y obeys $y \geq Cn^{1.2}rlogn$ for some positive numerical constant C, then with very high probability, most $n \times n$ matrices of rank r can be perfectly recovered. The convex program finds the matrix with the minimum nuclear norm that fits the data. This work is relevant to us. However, we do not have the luxury of uniformly sampled entries, instead, we are working with block-structured data.

[7] focuses on nuclear norm minimization to complete a matrix from partial entries. The authors show that their algorithm can recover an arbitrary $n \times n$ matrix of rank r from $O(nrlog^2(n))$ revealed entries, provided that revealed entries are drawn proportionally to the local row and column coherences (closely related to leverage scores) of the underlying matrix. They further propose a two-phase sampling algorithm that can perform nearly as well as local-coherence sampling without requiring a priori knowledge of the matrix coherence structure where the first phase uses uniformly sampled entries to estimate the coherence. However, we do not have the luxury of uniformly sampled entries. The algorithms developed in our paper make use of local coherence, albeit with a different goal of identifying the most informative assessor once the most informative candidate is identified.

3 PROBLEM SETTING AND PRELIMINARIES

We consider a scenario where $N_O \in \mathbb{N}$ candidates need to be ranked by a set of $N_A \in \mathbb{N}$ assessors who are partitioned into

P disjoint sets/panels. Each panel $p \in [P]$ assesses a set of candidates $S_p \subsetneq [N_O]$, where the candidate sets form a partition i.e., $\cup_p S_p = [N_O]$ and $\forall_{p \neq q} S_p \cap S_q = \emptyset$. Let $\mathbf{R} \in \mathbb{R}_+^{N_O \times N_A}$ be the matrix containing the ratings given by the panels to the candidates.

Goal: The goal of the learner is to use $\mathbf{R}$ and a small budget $B \in \mathbb{N}$ of extra (candidate, assessor) pairs that can be *actively/adaptively* queried to output a *good* learned ranking over the candidates, especially top few ranks.

3.1 Performance Measure:

Let $\mathbf{R}^* \in \mathbb{R}_+^{N_O \times N_A}$ be the true underlying rating matrix. Let σ^* be the ranking/permutation obtained by sorting the row sums of $\mathbf{R}^*$ in decreasing order. We measure the performance of any algorithm that outputs a ranking σ using the following metrics:

(i) **NDCG** is a normalization of the Discounted Cumulative Gain (DCG) measure widely used in practice [18]. DCG is a weighted sum of the degree of relevancy of the ranked items. The weight is a decreasing function of the rank (position) of the candidate and is therefore called a discount. The discount factor negatively impacts highly relevant items when they are ranked low. NDCG normalizes DCG by the Ideal DCG (IDCG), which is simply the DCG measure of the best ranking result. Thus NDCG measure is always a number in [0, 1]. The logarithmic discount $\frac{1}{\log_2(i+1)}$, where i is the candidate's rank, dominated the literature and applications. To focus on the top k ranks the discount is set to zero for ranks larger than k. Such NDCG measure is usually referred to as NDCG@k. NDCG is computed as,

$$NDCG = \frac{DCG}{IDCG} \text{ where } DCG = \sum_{i=1}^{N_O} \frac{g_i}{\log_2(i+1)} \tag{1}$$

where g_i is a relevance score of the i^{th} ranked candidate.

(ii) **Quality@k**: In many settings, it is crucial to obtain the top k ranked items accurately, regardless of their relative ranking order. We define *Quality@k* as the percentage of predicted top k candidates that are also present in true top k ranking.

3.2 Preliminaries

Matrix Completion: As every candidate is evaluated only by its panel and not by the set of all assessors, $\mathbf{R}$ has multiple unknown (candidate, assessor) pair values. There are many approaches to predict these unknown values using matrix completion [3–5, 9, 10, 15]. Matrix completion concerns recovering/predicting missing values in a matrix from a subset of its revealed entries. Most of the matrix completion algorithms recover a low-rank approximation of the given partially filled matrix. Nuclear norm minimization (NNC) [4, 15] is an effective method of recovering such a low-rank matrix. It gets *low-rank approximation* of a partially filled matrix, in our case $\mathbf{R}$ by optimizing the following convex problem,

$$\min_R \|R\|_* \; s.t. R_{ij} = R^*_{ij}, \forall (i,j) \in \Omega \tag{2}$$

Where $\mathbf{R}^*$ is an underlying true matrix, Ω is a set of all known entries, and nuclear norm $||.||_*$ of a matrix is a sum of its singular values. Nuclear norm minimization and similar methods assume that *(a)* the observed elements are randomly and uniformly chosen and *(b)* the underlying low-rank matrix is incoherent *i.e* it prevents the information of the row and column spaces of the matrix from being too concentrated in a few rows or columns.

Local Coherence: If the matrix has row and column space with low coherence, then each entry is expected to provide the same information. Consider the standard $rank-1$ coherent matrix example stated in [4]; The matrix $X = e_1 e_n^T$ where e_i is the i -th canonical basis vector in euclidean space has a single non-zero entry at $(1, n)$.

$$X = e_1 e_n^T = \begin{bmatrix} 0 & 0 & \dots & 0 & \mathbf{1} \\ 0 & 0 & \dots & 0 & 0 \\ . & . & \dots & . & . \\ . & . & \dots & . & . \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

Evidently, all the entries of X do not have the same information. hence it becomes crucial the way we sample/query unknown entries of such a coherent matrix.

In order to workaround assumption *(b)*, [7] proposed a *Two-phase sampling strategy*. They show us that the incoherence requirement can be eliminated, provided the sampling probability of an element being observed is dependent on the sum of the corresponding row and column leverage scores (local coherence parameters) of the underlying matrix. In the first phase, part of a given budget is used to query (candidate, assessor) pairs uniformly at random. The queried pairs help estimate *local coherence* parameters as shown below,

$$u_i = \frac{N_O}{r} \left\| U^T e_i \right\|^2, \; v_j = \frac{N_A}{r} \left\| V^T e_j \right\|^2 \tag{3}$$

where rank-r *singular value decomposition* (SVD) of $\mathbf{R}$ given by $U \Sigma V^T$, u_i is a local coherence of row i and v_j is a local coherence of column j. $\left\| U^T e_i \right\|^2$ and $\left\| V^T e_j \right\|^2$ are also referred as *leverage scores* in matrix sparsification literature [1, 8, 14]. These parameters are used to estimate local coherence distribution as demonstrated by the equation below,

$$P_{i,j} = \min \left\{ c_0 \frac{(u_i + v_j) m \log^2(N_O + N_A)}{\min\{N_O, N_A\}}, 1 \right\} \tag{4}$$

Estimated distribution helps to actively query (candidate, assessor) pair till one is out of budget.

4 MODELLING ASSUMPTIONS

We now present the exact assumptions we make on the underlying matrix.

`Assumption 1`: Ground Truth Matrix Structure : The ground truth matrix $\mathbf{R}^* \in \mathbb{R}_+^{N_O \times N_A}$ where each assessor reviews every candidate is assumed to be a *low-rank* matrix. In particular, we assume that each assessor evaluates a candidate based on the m features *i.e.* the rank of $\mathbf{R}^*$ is m, and it can be factored as $F * W^T$,

here $*$ denotes matrix multiplication. $F \in \mathbb{R}^{N_O \times m}$ captures the assumption that each candidate has a different expertise level for each feature and $W \in \mathbb{R}^{N_A \times m}$ contains the weight given by the assessor to each feature. As is common in matrix completion literature, we will assume that the algorithm knows the rank (or an upper bound) of the ground truth matrix. Based on the assumption, we consider two models,

- **Model** 1**:** We assume that i-th column of W follows a multivariate Gaussian distribution with mean $\mu_i \in \mathbb{R}^m$ and covariance $\sigma^2\mathbb{I}$.
- **Model** 2**:** This model is inspired by [11] which makes the following assumption:
  Base Factor Assumption: We assume that there exist m feature-expertise *base factor* vectors i.e., rows of a matrix F such that the rest of the rows/feature-expertise vectors can be obtained as some convex combination of these m vectors. Similarly, we assume that there exist m feature-weight base factor vectors *i.e.* rows of a matrix W such that the rest of the columns/feature-weight vectors can be obtained as some convex combination of these m vectors.

Remark: The Model 2 is motivated by the candidate-assessor scenario. The assumption made by [11] can be interpreted as the existence of a small set of prototype candidates and assessors whose combinations result in other candidates and assessors, respectively.

Proposition 1. *Under the Model* 2 *(base factor model), the best-ranked candidate (rank* 1*) will always be present in base factors.*

Proof: Say $f_1, f_2, ..., f_n$ are rows of F and $w_1, w_2, ..., w_n$ are rows of W, then the score of candidate i is $f_i(\sum_k w_k^T)$. The term $(\sum_k w_k^T)$ is common for all candidates. Thus any candidate whose row is a convex combination of base factors will have a score as a convex combination of scores of base factor candidates and so the maximum is attained by a candidate present in base factors.

`Assumption 2:` Availability of extra budget : We assume that we have a small extra budget $B \in \mathbb{N}$ to actively query (candidate, assessor) pairs in addition to the block diagonal entries that are already known.

5 ALGORITHMS

The main goal of this section is to present algorithms that efficiently utilize the budget adaptively to get close to the candidate's actual ranking. Towards this, we propose two algorithms `1. Query by Candidate Probability-Local coherence Probability (OPLP-Query)` and `2. Query by Base Factors - Local coherence Probability (BFLP-Query)` with different strategies to actively query (candidate, assessor) pairs. In the below sections, we will discuss these algorithms in detail.

5.1 Algorithm: `OPLP-Query`

We propose the algorithm `OPLP-Query` as shown in Algorithm 1 for the case where the ground truth rating matrix satisfies assumption 1 described in Section 4. The algorithm first identifies the candidate to query, followed by an assessor who will assess the candidate. The algorithm maintains a probability distribution over candidates where a higher probability is assigned to candidates with better-estimated rank. As the algorithm accumulates information about a currently higher-ranked candidate, the estimated score of the candidate will get closer to its actual score. If the selected candidate is truly top-ranked, it will continue to stay at the top of the estimated ranking as well. If not, eventually its rank will slide down closer to its actual rank.

Specifically, let $P_{\mathrm{O}} \in [0,1]^{N_O}$ be the vector of probability scores for candidates where the probability for the i^{th} candidate P_{O}^i is proportional to the average of the assessor scores given to the candidate. The algorithm first samples an candidate o_{id} using P_{O}. Next, it needs to select the assessor for the sampled candidate's assessment. To do this, the algorithm calculates the local coherence probability matrix (P_{LC}) using Algorithm 2. We concentrate on the selected candidate (o_{id}) row and normalize the local coherence scores to obtain the assessor probability vector P_{A}. It then samples an assessor a_{id} using P_{A} who has not assessed o_{id}, the chosen candidate before. The algorithm then queries the pair $(o_{\mathrm{id}}, a_{\mathrm{id}})$ and updates score in $\mathbf{R}$.

Algorithm 1 OPLP-Query

Input: Block-diagonal matrix $\mathbf{R} \in \mathbb{R}_+^{N_O \times N_A}$ budget: $\mathbf{B}$.
while $\mathbf{B} > 0$ **do**
  Find the mean score of each candidate.

$$\forall i \; M_i = \frac{\sum_{j=1}^{N_A} R_{i,j}}{\sum_{j=1}^{N_A} \mathbb{1}(R_{i,j} != 0)}$$

  Find the candidate probabilities,

$$\forall i \; P_{\mathrm{O}}[i] = \frac{M_i}{\sum_{j=1}^{N_O} M_j}$$

  $o_{\mathrm{id}} \leftarrow$ Sample candidate based on P_{O}.
  Find the local coherence probability matrix (P_{LC}) for $\mathbf{R}$ using Algorithm 2.
  Find assessor probabilities for the sampled candidate (o_{id}) row,

$$\forall i \; P_{\mathrm{A}}[i] = \frac{P_{\mathrm{LC}}[o_{id}][i]}{\sum_{j=1}^{N_A} P_{\mathrm{LC}}[o_{id}][j]}$$

  $a_{\mathrm{id}} \leftarrow$ Sample assessor according to P_{A}
  Query $(o_{\mathrm{id}}, a_{\mathrm{id}})$ pair and update matrix $\mathbf{R}$
  Update available budget, $\mathbf{B} = \mathbf{B} - 1$
end while
$Z \leftarrow$ Complete matrix using Nuclear norm minimization by solving Equation 2.
Evaluate `average score` of the candidate.

$$\forall i \; M_i = \frac{\sum_{j=1}^{N_A} Z_{i,j}}{N_A}$$

$\sigma \leftarrow$ Rank the candidates based on the decreasing `average score` M.
Output: $\sigma \leftarrow$ Final ranking of the candidates

Algorithm 2 Compute Local Coherence Matrix (Ref. [7])

Input: Round 1 evaluation matrix $\mathbf{R} \in \mathbb{R}^{N_O \times N_A}$, rank parameter m

Initialize: Universal constant $c_0 = 0.1$

1 : Compute the rank-m SVD of $\mathbf{R}$, $U\Sigma V^T$

2 : Estimate the local coherence's by $u_i = \frac{N_O}{m} \left\|U^T e_i\right\|^2$ and $v_i = \frac{N_A}{m} \left\|V^T e_j\right\|^2$

3 : Calculate the local coherence probability for $(candidate_i, assessor_j)$ pair as,

$$P_{i,j} = \min\left\{c_0 \frac{(u_i + v_j) m \log^2(N_O + N_A)}{\min\{N_O, N_A\}}, 1\right\}$$

Output: Local coherence probability matrix P_{LC}

To calculate the candidate probabilities for the subsequent query, use the updated $\mathbf{R}$ as input. Once the algorithm runs out of budget, it completes the updated matrix $\mathbf{R}$ by solving the nuclear norm minimization problem presented in Equation 2. We used the nuclear norm minimization algorithm of [2] to complete the updated matrix $\mathbf{R}$. However, any matrix completion sub-routine can be used for the same. Let Z be the completed matrix. We find the average score vector M of candidates using Z, and rank the candidates as per their average scores.

Remark: The main idea is that once the candidate who is most likely the top ranked is selected, the assessor who will give the most *information* to validate the candidate's choice is chosen. A natural way to quantify this information is using local coherence. Note that the chosen pair may not necessarily be the one that has the highest local coherence in the matrix. As the algorithm does not need to complete the *entire* matrix reliably but only needs to get good rankings at the top, it queries for the pair that gives the most information, given that the candidate is likely to be at the top of the ranking.

5.2 Algorithm: `BFLP-Query`

We propose a different algorithm `BFLP-Query` shown in Algorithm 3. It assumes that the ground truth follows Model 2 described in Section 4. We describe this algorithm below.

BFLP-Query starts by completing the incomplete input matrix $\mathbf{R}$ using nuclear norm minimization [2]. Let Z be the completed matrix. Let S be the simplex form of Z, where each row i of S satisfies $\{S_i \in [0,1]^{N_A} : \|S_i\|_1 \leq 1\}$. According to the assumption of Model 2, there are m *base row factors* which the algorithm first attempts to find. To do this, the algorithm partially makes use of the approach developed in [11]. The goal of [11] was to eliminate rows and columns to obtain the entry with the maximum value in a matrix that follows Model 2. However, we wish to rank entries based on row sums. Thus, BFLP-Query does *only* row elimination to find a set of prospective candidates. Let $\Pi_m([N_O])$ be the set of all m subsets of N_O candidates. From all possible $m-$subsets, the algorithm identifies I^* - a prospective set of *base row factors* of S. Proposition 1

Algorithm 3 BFLP-Query

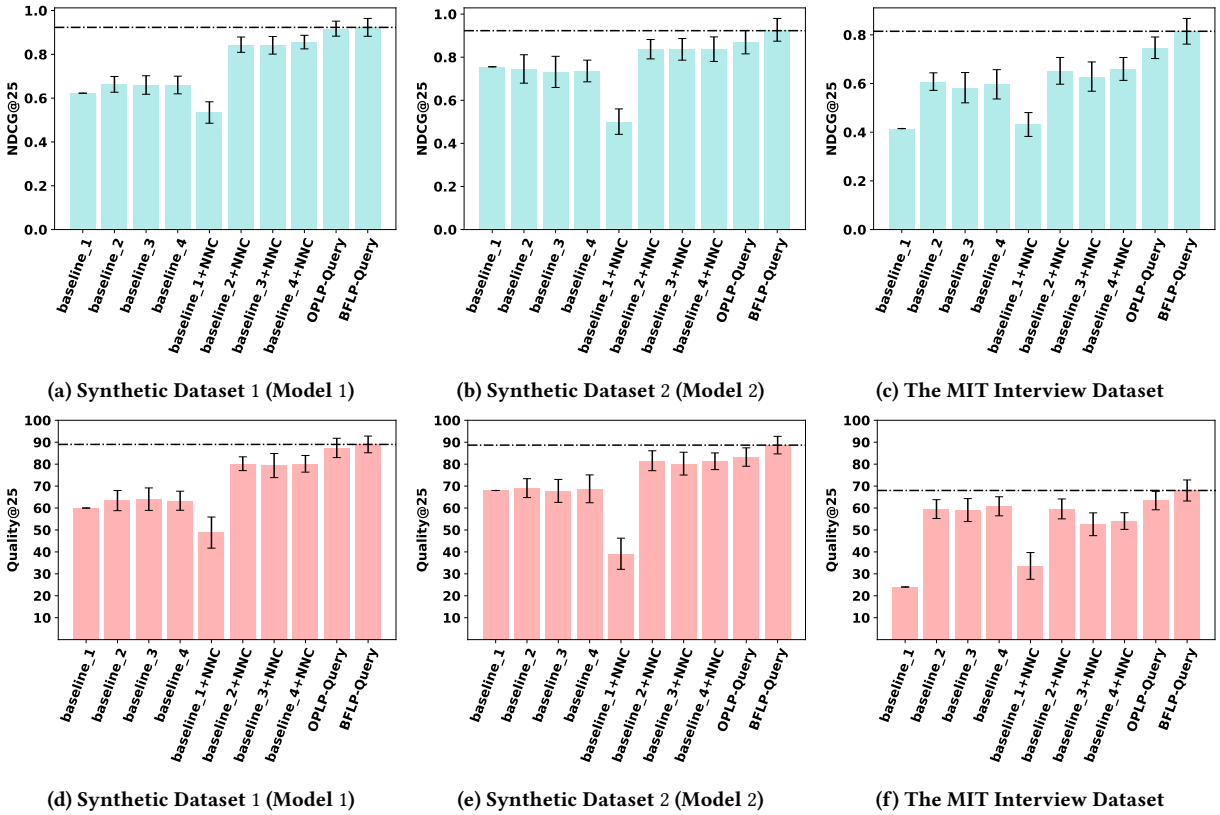
Input: Block-diagonal matrix $\mathbf{R} \in \mathbb{R}^{N_O \times N_A}$, budget: $\mathbf{B}$, rank parameter: m
$Z \leftarrow$ Complete input matrix $\mathbf{R}$ by solving NNC Equation 2.
while $\mathbf{B} > 0$ **do**
  $S \leftarrow$ Simplex form of matrix Z, where i^{th} row of S, S_i is $\{S_i \in [0,1]^{N_A} : \|S_i\|_1 \leq 1\}$
  $I_1 \leftarrow$ Choose any $m-$rows from S
  $J_1 \leftarrow$ Choose any $m-$columns from S
  $I^* = I_1$
  for all $I \in \Pi_m([N_O])$ **do**
    if $det^2(S(I, J_1)) > det^2(S(I^*, J_1))$ **then**
      $I^* = I$
    end if
  end for
  $P_{\text{LC}} \leftarrow$ Calculate local coherence probabilities for the candidates using Algorithm 2.
  repeat
    $(o_{\text{id}}, a_{\text{id}}) \leftarrow$ Sample 1 entry from I^* using local coherence probabilities.
  until $(o_{\text{id}}, a_{\text{id}})$ pair not queried before
  Query $(o_{\text{id}}, a_{\text{id}})$ and update matrix $\mathbf{R}$
  Update available budget. $\mathbf{B} = \mathbf{B} - 1$
  $Z \leftarrow$ Complete input matrix R using NNC by solving Equation 2.
end while
Calculate the average score of the candidate

$$\forall i \; M_i = \frac{\sum_{j=1}^{N_A} Z_{i,j}}{N_A}$$

$\sigma \leftarrow$ Ranking the candidate based on the decreasing $score$.
Output: $\sigma \leftarrow$ Final ranking of the candidates

shows that if data follows Model 2, then the best-ranked candidate necessarily must belong to I^*, thus making it a good set of candidates to zoom into. Once a smaller set I^* of candidates is identified, similar to OPLP-Query, the BFLP-Query algorithm calculates local coherence probability score matrix $P_{\text{LC}} \in [0,1]^{N_O \times N_A}$ using Algorithm 2. It then samples one $(candidate, assessor)$ pair according to the computed local coherence probabilities where the candidate belongs to I^* and a_{id} has not assessed o_{id} in previous rounds. It queries $(o_{\text{id}}, a_{\text{id}})$ pair and updates the scores in $\mathbf{R}$ and completes the updated matrix. The procedure is repeated till the algorithm runs out of budget. The final ranking of candidates is determined by computing the average score of each candidate *i.e* averaging the rows using Z.

Remark 1: In our setting, Assumptions required for the standard matrix completion (see Section 3.2) need not always hold. Figure 1 shows that matrix completion assumption *(a)* does not hold in our setting as samples are not uniformly and randomly chosen but have a disjoint diagonal block structure. Hence standard matrix completion techniques fail in our settings.



(a) Synthetic Dataset 1 (Model 1) (b) Synthetic Dataset 2 (Model 2) (c) The MIT Interview Dataset

(d) Synthetic Dataset 1 (Model 1) (e) Synthetic Dataset 2 (Model 2) (f) The MIT Interview Dataset

Figure 2: first row - Algorithms vs $NDCG@25$, Second row - Algorithms vs $Quality@25$: The average results after 100 iterations of the proposed algorithms are compared with a set of baseline algorithms along with *error* bars. *(baseline#+NNC)* is nuclear norm minimization over baseline#.

Remark 2: We note that the sampling strategy proposed in [7] does not directly help in our setting, primarily because the small budget does not allow us to have a two-phase sampling strategy. Moreover, if we use the block diagonal entries as a substitute for phase 1 and evaluate the local coherence score using the known entries along the block diagonal, there is a chance that the evaluated local coherence score is inaccurate, as the sampling requirement stated in [7] requires the entries to be sampled uniformly at random which is not true in our case. Furthermore, and perhaps more importantly, the goal of [7] is to complete the entire matrix accurately, whereas our goal is to complete it well enough so that the obtained rankings are good at the top of the list.

6 EXPERIMENTS

We now present the experimental setup, performance metrics, and results on different datasets. We have tested our Algorithms on two synthetic datasets in Section 6.2 and The MIT interview real-world dataset in Section 6.4.

Performance Metric:

(i) `NDCG@k` *(Normalized Discounted Cumulative Gain)* measures the ranking quality of the algorithm by penalizing inaccurately ranked candidates. We used Equation 1 to calculate *NDCG* score.

(ii) `Quality@k`:The Percentage of predicted top k candidates that are also present in true top k ranking.

6.1 Baseline Algorithms:

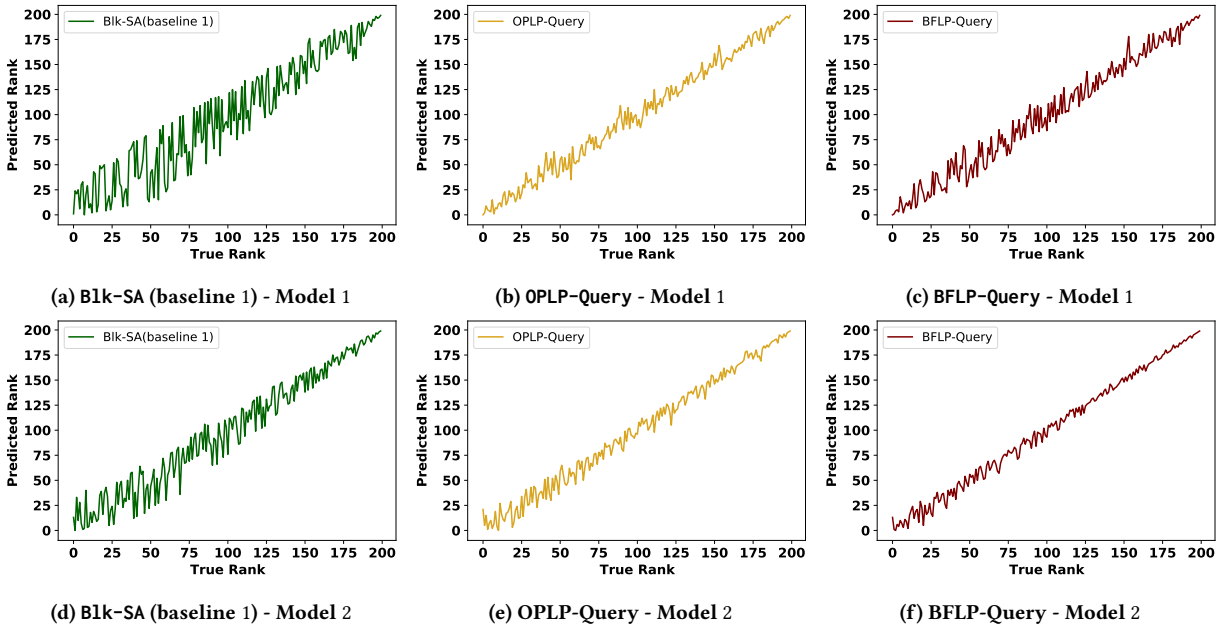
We compare our results with,

(i) `Blk-SA` (baseline 1): Final score of a candidate is the average score given by the respective candidate's panel.

(ii) `Blk-rand-Bgt` (baseline 2) : Actively query budget $B \in \mathbb{N}$ (candidate, assessor) pairs uniformly.

(iii) `Blk-Lc-Bgt` (baseline 3) : Using equation (3) calculate the local coherence probability score [7] for each (candidate, assessor) pair and sample **B** pairs using these probabilities. Along with the panel's initial assessment score, newly queried entries were used to calculate the average score of the candidate.

(iv) `Blk-rand-Lc-Bgt` (baseline 4) : Query $\frac{\mathbf{B}}{2}$ (candidate,assessor) pairs uniformly, and $\frac{\mathbf{B}}{2}$ pairs using local coherence probability.



(a) Blk-SA (baseline 1) - Model 1 (b) OPLP-Query - Model 1 (c) BFLP-Query - Model 1

(d) Blk-SA (baseline 1) - Model 2 (e) OPLP-Query - Model 2 (f) BFLP-Query - Model 2

Figure 3: *Predicted Rank vs True Rank*: The plot shows the deviation of the Predicted rank from the True rank for the, *first row* - synthetic dataset 1(Model 1), *second row* - synthetic dataset 2 (Model 2).

In addition, we applied the matrix completion technique on each baseline algorithm mentioned above to predict scores of $(candidate_i, assessor_j)$ pair where $assessor_j$ has not scored $candidate_i$. We used Nuclear Norm Minimization based matrix completion method for our experiments [2, 4, 15].

6.2 Data generation of Synthetic Datasets

Experimental Setup: We considered $N_O = 200$ candidates assessed by five panels. Each panel has six assessors, and each candidate is evaluated based on m features.

Data Generation for Synthetic Dataset 1: In this experiment, we generate data according to model 1. candidates are assigned a skill score for each feature, where scores are chosen from $Unif([1, 10])$. Let $F_{200\times m}$ denote a candidate's skill score for each feature. Let $W_{30\times m}$ denote a weight that each assessor gives to each feature. The i-th column of W is sampled from $\mathcal{N}(\mu_i, \sigma^2\mathbb{I})$ where we fix $\sigma = 3$ in our experiments. μ for each feature is chosen from $Unif([1, 10])$. $\mathbf{R}^* = FW^T$ indicates the final score given by each assessor to each candidate when every assessor assesses every candidate. We assume $\mathbf{R}^*$ as a low-rank matrix of rank 3. $\mathbf{R}^*$ is a ground truth complete matrix. In our scenario, the candidate is not assessed by each assessor but by its panel members only; hence, we will mask $(candidate_i, assessor_j)$ pair of $\mathbf{R}^*$ when $assessor_j$ has not assessed $candidate_i$. After masking such pairs, we get our block diagonal structured assessment data $\mathbf{R} \in \mathbb{R}^{200\times 30}$.

Data generation for Synthetic Dataset 2: In this experiment, we generate data according to model 2. Here again, $\mathbf{R}^* = F * W^T$ where $F \in \mathbb{R}^{200\times m}$ and $W \in \mathbb{R}^{30\times m}$. To generate F and W, we selected m candidates from $Unif([1, 200])$ and assigned skill score for m features sampled from the $Unif([1, 10])$. These are the *base row factors*. We then generate $200-m$ rows by taking random *convex combinations* of these base row factors. We generate W similarly.

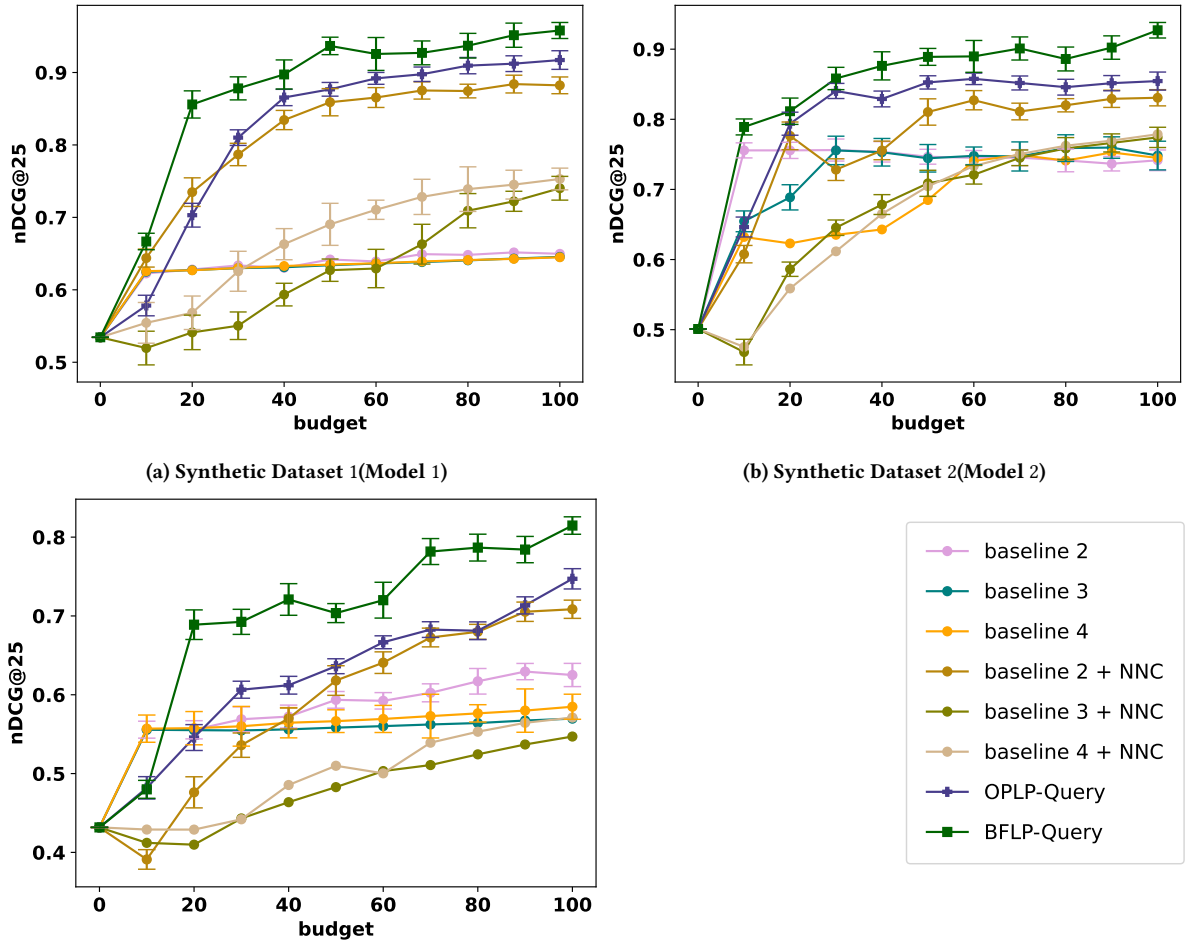
6.3 Analysis of Results over Synthetic Datasets

Comparison with baseline Algorithms : Figure 2a, 2b shows the comparison of the several baseline algorithms considered and the proposed algorithms for the *nDCG@25* metric and the Figure 2d, 2e show the comparison of the several baseline algorithms considered and the proposed algorithms for the *Quality@25* metric. As can be seen, just having an extra budget does not help to improve the rankings. Choosing meaningful (candidate, assessor) pairs is crucial, which helps obtain good rankings. Both the proposed algorithms produce superior results than all the baseline algorithms considered.

Figure 2 shows that standard matrix completion (baseline 1 + NNC) fails in our settings as assumptions required for the standard matrix completion (see Section 3.2) need not always hold. Figure 1 shows that matrix completion assumption *(a)* does not hold in our setting as samples are not uniformly and randomly chosen but have a disjoint diagonal block structure.

Effect of budget: We observe the Effect of budget on several algorithms in Figure 4. We can observe apparent performance enhancement when we have more budget in hand. It shows that the proposed algorithms perform better than just using budget by uniform and random sampling. In particular, we observe that *BFLP-Query* shows fast convergence than the rest of the algorithms.

Quality of Entire Ranking: Figure 3a and 3d show results for baseline 1. Error in predicted rankings is high in this case. It can



(a) Synthetic Dataset 1(Model 1)

(b) Synthetic Dataset 2(Model 2)

(c) The MIT Interview Dataset.

Figure 4: *nDCG@25 vs budget*: The plot shows the performance of the proposed algorithms and the Baseline algorithm for the different budgets in hand. (To maintain plot readability, error bars for top performing baselines showed in figure.)

be seen from Figure 3, error significantly reduces for our proposed algorithms `OPLP-Query` and `BFLP-Query`. Deviation of predicted rank from the true rank is significantly lesser for all other baselines as well. The results for the other baselines can be found in the GitHub repository mentioned in Section 8.

6.4 Experiments on "MIT Interview Dataset" (Real Data):

The MIT Interview Dataset [13] has audio-video recordings of 138 mock job interviews conducted by professional career counselors of 69 undergraduate students. Every student gave two different interviews *i.e* total of 138 interviews. The dataset has Amazon Mechanical Turk ratings of each interview, where human annotators judged each interview based on 16 different features. Total of nine assessors judged each student. The total score given by the assessor is the sum of the score given to each feature. Data collection and evaluation procedure is mentioned in [13]. In this dataset, each interviewer assessed every candidate *i.e* we have $\mathbf{R}^* \in \mathbb{R}_+^{138\times 9}$. However, in practice, the assessors are typically divided into panels and each panel assesses a subset of candidates. To capture this scenario, we divided the 9 assessors into 4 disjoint panels such that three panels have 2 panelists each and one panel has 3 panelists. We assigned two panels to 34 students, and the other two panels were assigned to 35 students each. We refer to this block diagonal matrix as $\mathbf{R}$. All our algorithms were given $\mathbf{R}$ as input and were allowed to query additional (candidate, assessor) pairs not in $\mathbf{R}$ but in $\mathbf{R}^*$. We compare the predicted rankings with the actual rankings obtained from the $\mathbf{R}^*$ using `NDCG` and `Quality` metrics. We now present the results of this dataset.

Comparison with baseline Algorithms : Figure 2c shows the comparison of `OPLP-Query` and `BFLP-Query` with baseline algorithms based on *nDCG@25* metric and Figure 2f shows the comparison of proposed algorithms with baseline algorithms based on *Quality@25* metric. We assumed to have an extra budget of

Table 1: Effect of Hyper Parameter k : Experimental results based on the top 20 candidate rankings(Smaller k) and top 50 candidate rankings(Higher k). *(baseline#+NNC)* is nuclear norm minimization over baseline#.

	Results for k = 20 (Smaller k)				**Results for k = 50 (Higher k)**			
	Synthetic Dataset 1		**Synthetic Dataset 2**		**Synthetic Dataset 1**		**Synthetic Dataset 2**	
Algorithm	Quality @20(%)	NDCG @20	Quality @20(%)	NDCG @20	Quality @50(%)	NDCG @50	Quality @50(%)	NDCG @50
Blk-SA (baseline 1)	40.00	0.3419	68.00	0.4796	60.00	0.6226	68.00	0.755
Blk-rand-Bgt (baseline 2)	48.55	0.3925	69.30	0.4999	63.40	0.6630	69.08	0.7454
Blk-Lc-Bgt (baseline 3)	50.55	0.4046	68.10	0.4922	64.04	0.669	67.80	0.7318
Blk-rand-Lc-Bgt (baseline 4)	49.30	0.3959	69.40	0.4959	63.33	0.661	68.76	0.736
baseline 1 + NNC	44.50	0.3615	47.50	0.3472	48.80	0.534	39.16	0.5010
baseline 2 + NNC	74.50	0.6012	71.43	0.5234	82.20	0.884	81.56	0.837
baseline 3 + NNC	73.65	0.5970	71.83	0.5248	79.36	0.8612	80.24	0.8364
baseline 4 + NNC	74.81	0.6089	72.01	0.5235	81.16	0.8759	81.32	0.8372
OPLP-Query	**79.60**	**0.6580**	**76.23**	**0.5796**	**86.40**	**0.9172**	**83.64**	**0.8545**
BFLP-Query	**81.20**	**0.6650**	**80.43**	**0.6021**	**89.00**	**0.9231**	**88.67**	**0.9270**

Table 2: The baselines show notable discrepancies in ranking the top ten candidates on the MIT Interview data. However, the algorithms OPLP-Query and BFLP-Query outperform these baselines. OPLP-Query places just two candidates beyond the 25th position, while BFLP-Query successfully includes all true top 10 candidates in the predicted top 20 list. *(baseline#+NNC)* is nuclear norm minimization over baseline#.

Algorithm	**Analysis of predicted ranks of true top 10 candidates**
Baseline 1	**Five** candidates are ranked beyond **40th** position.
Baseline 2	**Four** candidates are ranked beyond **60th** position.
Baseline 3	**Four** candidates are ranked beyond **60th** position.
baseline 4	**Four** candidates are ranked beyond **70th** position.
baseline 1 + NNC	**Three** candidates are ranked beyond **100th** position.
baseline 2 + NNC	**Four** candidates are ranked beyond **25th** position (One ranked beyond 75)
baseline 3 + NNC	**Three** candidates are ranked beyond **40th** position.
baseline 4 + NNC	**Four** candidates are ranked beyond **25th** position (One ranked beyond 50)
OPLP-Query	**Two** candidates ranked beyond **25th** position.All other ranked under **20th** position
BFLP-Query	**All** true top 10 candidates lie in predicted **top** 20 list

B = 70 for this set of experiments. Figure 2c and Figure 2f shows that `OPLP-Query` and `BFLP-Query` clearly outperform baseline algorithms which do not use extra *budget* and it is marginally better than other baseline algorithms having the luxury of extra budget and `BFLP-Query` outperforms all the baseline algorithms considered.

Need for Extra budget : Figure 5 shows that the widely used baseline algorithm `Blk-SA` (*Simple Averaging*) ranks only 30% of true top 10 candidates accurately. Another baseline algorithm `baseline1+ NNC` which does not use any extra budget but completes the matrix using only block diagonal entries do not show any improvement and it also ranks only 30% of true top 10 candidates accurately. While `OPLP-Query` ranks 50% of true top 10 candidates accurately. `BFLP-Query` clearly outperforms these baseline algorithms and ranks 70% of true top 10 candidates accurately.

Figure 5 shows that the predicted ranks by baseline algorithms - `Blk-SA` (baseline 1) and `baseline1+ NNC`, are even beyond rank 50 for true top 10 candidates. While every student in the true top 10 rank list is present in the predicted top 20 rank list of `BFLP-Query`. Moreover, most of the candidates predicted by *OPLP-Query* are in the top 20 list. Table 2 compares results for all the baselines. We can observe that the proposed algorithms drastically reduce the deviation of the predicted rankings from the actual rankings.

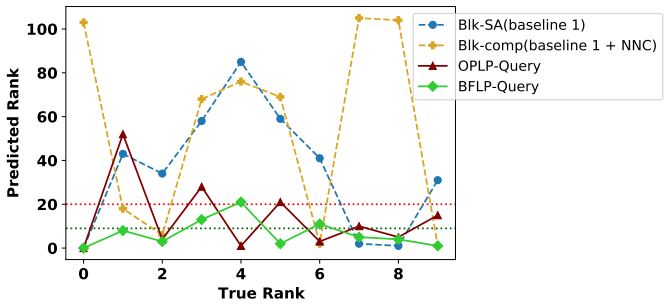


Figure 5: Experiment on MIT Interview Dataset : (*Predicted Rank vs True Rank*): The plot shows the performance of the different baseline and proposed algorithms to retrieve True Top 10 Candidates. All candidates below *dotted Green* line are predicted correctly in the Top 10. Candidates below *dotted red* line and above *dotted green* line are close to their actual ranking as they ranked below rank 20. Candidates above *dotted red* line are considered to be ranked poorly.

6.5 Further Experiments

Effect of Higher Number of Candidates: We analysed results when number of candidates are higher. We evaluated the proposed algorithms for the scenario where 500 candidates are assessed by 30 assessors. This is a 2.5 times larger matrix in terms of number of entries than the $200 * 30$ reported in the Section 6.2. Results shows that proposed algorithms outperforms baselines for scenarios where a higher number of candidates needs to be evaluated. The results can be found in the GitHub repository mentioned in Section 8.

Effect of Hyper parameter k***:***

Smaller k. : We observe the results for the proposed methods when $k = 20$. The Table 1 displays the performance evaluation of several algorithms based on the top 20 candidate rankings which clearly demonstrates that the proposed methods perform better than the baselines for smaller value of k.

Higher k. : We examined the outcomes of the proposed approaches at a value of $k = 50$. The performance evaluation of various algorithms, considering the top 50 candidates, is presented in Table 1. These results demonstrate that the proposed algorithms surpass several baseline methods, even when evaluating performance at the top 50 candidate rankings (with a higher value of k).

7 CONCLUSION

We considered the problem of ranking a set of candidates which are rated by a disjoint set of assessors under two natural models for human assessors. We proposed two novel algorithms for the same, and our extensive experimental evaluation shows the efficacy of the proposed algorithms. Future work includes showing formal guarantees for the extra budget needed to obtain reliable rankings. Another interesting direction is to consider the multi-armed bandits variant of this problem where the same (candidate, assessor) pair might be sampled multiple times to get stochastic outcomes. However, unlike the standard bandit's setup, the goal here would be to identify the top k candidates, which would make it a challenging problem.

8 CODES FOR REPRODUCING RESULTS

The datasets and codes are available here.

REFERENCES

[1] Christos Boutsidis, Michael W Mahoney, and Petros Drineas. 2009. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 968–977.
[2] Ricardo Cabral, Fernando De la Torre, João P Costeira, and Alexandre Bernardino. 2013. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *Proceedings of the IEEE international conference on computer vision*. 2488–2495.
[3] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization* 20, 4 (2010), 1956–1982.
[4] Emmanuel Candes and Benjamin Recht. 2012. Exact matrix completion via convex optimization. *Commun. ACM* 55, 6 (2012), 111–119.
[5] Emmanuel J Candes and Yaniv Plan. 2010. Matrix completion with noise. *Proc. IEEE* 98, 6 (2010), 925–936.
[6] Shayok Chakraborty, Jiayu Zhou, Vineeth Balasubramanian, Sethuraman Panchanathan, Ian Davidson, and Jieping Ye. 2013. Active matrix completion. In *2013 IEEE 13th international conference on data mining*. IEEE, 81–90.
[7] Yudong Chen, Srinadh Bhojanapalli, Sujay Sanghavi, and Rachel Ward. 2014. Coherent matrix completion. In *International Conference on Machine Learning*. PMLR, 674–682.
[8] Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. 2012. Fast approximation of matrix coherence and statistical leverage. *The Journal of Machine Learning Research* 13, 1 (2012), 3475–3506.
[9] David Gross. 2011. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory* 57, 3 (2011), 1548–1566.
[10] Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. 2010. Matrix completion from a few entries. *IEEE transactions on information theory* 56, 6 (2010), 2980–2998.
[11] Branislav Kveton, Csaba Szepesvári, Anup Rao, Zheng Wen, Yasin Abbasi-Yadkori, and S Muthukrishnan. 2017. Stochastic low-rank bandits. *arXiv preprint arXiv:1712.04644* (2017).
[12] Robert S MacKay, Ralph Kenna, Robert J Low, and Sarah Parker. 2017. Calibration with confidence: a principled method for panel assessment. *Royal Society open science* 4, 2 (2017), 160760.
[13] Iftekhar Naim, M Iftekhar Tanveer, Daniel Gildea, and Mohammed Ehsan Hoque. 2015. Automated prediction and analysis of job interview performance: The role of what you say and how you say it. In *2015 11th IEEE international conference and workshops on automatic face and gesture recognition (FG)*, Vol. 1. IEEE, 1–6.
[14] Dimitris Papailiopoulos, Anastasios Kyrillidis, and Christos Boutsidis. 2014. Provable deterministic leverage score sampling. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 997–1006.
[15] Benjamin Recht. 2011. A simpler approach to matrix completion. *Journal of Machine Learning Research* 12, 12 (2011).
[16] Nihar B Shah. 2022. Challenges, experiments, and computational solutions in peer review. *Commun. ACM* 65, 6 (2022), 76–87.
[17] Jingyan Wang and Nihar Shah. 2019. Your 2 is My 1, Your 3 is My 9: Handling Arbitrary Miscalibrations in Ratings. In *AAMAS Conference proceedings*.
[18] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. 2013. A theoretical analysis of NDCG type ranking measures. In *Conference on learning theory*. PMLR, 25–54.