Budget-optimal Crowdsourcing using Low-rank Matrix Approximations

David R. Karger, Sewoong Oh, and Devavrat Shah
Department of EECS, Massachusetts Institute of Technology
Email: {karger, swoh, devavrat}@mit.edu

Abstract— **Crowdsourcing systems, in which numerous tasks are electronically distributed to numerous "information piece-workers", have emerged as an effective paradigm for human-powered solving of large scale problems in domains such as image classification, data entry, optical character recognition, recommendation, and proofreading. Because these low-paid workers can be unreliable, nearly all crowdsourcers must devise schemes to increase confidence in their answers, typically by assigning each task multiple times and combining the answers in some way such as majority voting.**

In this paper, we consider a model of such crowdsourcing tasks and pose the problem of minimizing the total price (i.e., number of task assignments) that must be paid to achieve a target overall reliability. We give a new algorithm for deciding which tasks to assign to which workers and for inferring correct answers from the workers' answers. We show that our algorithm, based on low-rank matrix approximation, significantly outperforms majority voting and, in fact, is order-optimal through comparison to an oracle that knows the reliability of every worker.

I. Introduction

Background. Crowdsourcing systems have emerged as an effective paradigm for human-powered problem solving and are now in widespread use for large-scale data-processing tasks such as image classification, video annotation, form data entry, optical character recognition, translation, recommendation, and proofreading. Crowdsourcing systems such as Amazon Mechanical Turk[1] establish a market where a "taskmaster" can submit batches of small tasks to be completed for a small fee by any worker choosing to pick them up. For example a worker may be able to earn a few cents by indicating which images from a set of 30 are suitable for children (one of the benefits of crowdsourcing is its applicability to such highly subjective questions).

Because the tasks are tedious and the pay is low, errors are common even among workers who make an effort. At the extreme, some workers are "spammers", submitting arbitrary answers independent of the question in order to collect their fee. Thus, all crowdsourcers need strategies to ensure the reliability of answers. Because the worker crowd is large, anonymous, and transient, it is generally difficult to build up a trust relationship with particular workers.[2] It is also difficult to condition payment on correct answers, as the correct answer may never truly be known and delaying payment can annoy workers and make it harder to recruit them to your task. Instead, most crowdsourcers resort to redundancy, giving each task to multiple workers, paying them all irrespective of their answers, and aggregating the results by some method such as majority voting.

For such systems there is a natural core optimization problem to be solved. Assuming the taskmaster wishes to achieve a certain reliability in their answers, how can she do so at minimum cost (which is equivalent to asking how she can do so while asking the fewest possible questions)?

Several characteristics of crowdsourcing systems make this problem interesting. Workers are neither persistent nor identifiable; each batch of tasks will be solved by a worker who may be completely new and who you may never see again. Thus one cannot identify and reuse particularly reliable workers. Nonetheless, by comparing one worker's answer to others' on the same question, it is possible to draw conclusions about a worker's reliability, which can be used to weight their answers to other questions in their batch. However, batches must be of manageable size, obeying limits on the number of tasks that can be given to a single worker.

Another interesting aspect of this problem is the *choice of task assignments*. Unlike many inference problems which makes inferences based on a fixed set of signals, we can choose which signals to measure by deciding which questions to ask to which workers. This makes designing a crowdsourcing system challenging in that we are required to determine how to allocate the tasks as well as design an algorithm to infer the correct answers once the workers submit their answers.

In the remainder of this introduction, we will define a formal model that captures these aspects of the problem. Then, we will describe how to allocate tasks and infer the correct answers. We subsequently describe how these two procedures can be integrated into a single Budget-optimal Crowdsourcing algorithm. We show that this algorithm is order-optimal: for a given target error rate, it spends only a constant factor times the minimum necessary to achieve that error rate.

Setup. We model a set of m tasks $\{t_i\}_{i\in[m]}$ as each being associated with an unobserved 'correct' solution $s_i \in \{\pm 1\}$. Here and after, we use $[N]$ to denote the set of first N integers. In the image categorization example stated earlier, tasks corresponds to labeling m images as suitable for children $(+1)$ or not (-1). These m tasks are assigned to n workers from the crowd. We use $\{w_j\}_{j\in[n]}$ to denote this

[1]http://www.mturk.com

[2]For certain high-value tasks, crowdsourcers can use entrance exams to "prequalify" workers and block spammers, but this increases the cost of the task and still provides no guarantee that the workers will try hard after qualification.

set of n workers.

When a task is assigned to a worker, we get a possibly inaccurate answer from the worker. We use $A_{ij} \in \{\pm 1\}$ to denote the answer if task t_i is assigned to worker w_j. Some workers are diligent whereas other workers might be spammers. We choose a simple model to capture the presence of spammers, which we call the *spammer-hammer* model. Under this model, we assume that each worker is either a hammer or a spammer. A hammer always gives the correct answer to all the questions and a spammer always gives random answers. Each worker w_j is labeled by a reliability parameter $p_j \in \{1/2, 1\}$, such that $p_j = 1$ if w_j is a hammer and $p_j = 1/2$ if w_j is a spammer.

According to the spammer-hammer model, if task t_i is assigned to worker w_j then

$$\mathbf{A}_{ij} \;=\; \begin{cases} s_i & \text{with probability } p_j \;, \\ -s_i & \text{with probability } 1-p_j \;, \end{cases} \qquad (1)$$

and $\mathbf{A}_{ij} = 0$ if t_i is not assigned to w_j. The random variable $\mathbf{A}_{ij}$ is independent of any other event given p_j. (Throughout this paper, we use boldface characters to denote random variables and random matrices unless it is clear from the context.)

We further assume that each worker's reliability is independent and identically distributed, such that each worker is a hammer with probability q and spammer with probability $1-q$:

$$\mathbf{p}_j \;=\; \begin{cases} 1 & \text{with probability } q \;, \\ 1/2 & \text{with probability } 1-q \;. \end{cases}$$

It is quite realistic to assume the existence of such a prior distribution for $\mathbf{p}_j$'s. In particular, it is met if we simply randomize the order in which we upload our task batches, since this will have the effect of randomizing which workers perform which batches, yielding a distribution that meets our requirements. On the other hand, it is not realistic to assume that we know what the prior is. To execute our inference algorithm, we do not require the knowledge of the hammer probability q. On the other hand, q is necessary in deciding how many times a task should be replicated to achieve certain reliability, and we discuss a simple way to overcome this limitation in Section II-D.

Under this crowdsourcing model, a taskmaster first decides which tasks should be assigned to which workers, and then infer the correct solutions $\{s_i\}_{i\in[m]}$ once all the answers $\{\mathbf{A}_{ij}\}$ are submitted. We assume a *one-shot* model in which all questions are asked simultaneously and then an estimation is performed after all the answers are obtained. In particular, we do not allow allocating tasks adaptively based on the answers received thus far. Then, assigning tasks to nodes amounts to designing a bipartite graph $G(\{t_i\}_{i\in[m]} \cup \{w_j\}_{j\in[n]}, E)$ with m task nodes and n worker nodes. Each edge $(i,j) \in E$ indicates that task t_i was assigned to worker w_j.

With a slight abuse of notations, we use a matrix $\mathbf{A} \in \{0, 1, -1\}^{m\times n}$ to denote the randomly weighted adjacency matrix of the graph G: edge $(i,j) \in E$ is weighted with the submitted answer $\mathbf{A}_{ij} \in \{+1, -1\}$ and $\mathbf{A}_{ij} = 0$ if $(i,j) \notin E$. We shall use C_1, C_2, etc. to denote general constants. Whenever we say a property $\mathcal{A}$ holds with high probability (w.h.p), we mean that there exists a function $f(m,n)$ such that $\mathbb{P}(\mathcal{A}) \geq 1 - f(m,n)$ and $\lim_{m,n\to\infty} f(m,n) = 0$.

Prior Work. A naive approach to aggregate information from multiple workers is to use majority voting. Majority voting simply follows what the majority of workers agree on. When we have many spammers in the crowd, majority voting is error-prone since it gives the same weight to all the answers, regardless of whether they are from a spammer or a diligent workers. We will show in Section II-C that majority voting is provably sub-optimal and can be significantly improved upon.

To fully exploit redundancy, we need to infer the reliability of the workers simultaneously while inferring the solutions of the tasks. Dawid and Skene [DS79] proposed an iterative algorithm for inferring the solutions and reliability of workers, based on expectation maximization (EM) [DLR77]. EM is a heuristic inference algorithm that iteratively does the following: given workers' answers to the tasks, the algorithm attempts to estimate the reliability of the workers and given estimation of reliability (error probabilities) of workers, it estimates the solution of the tasks; and repeat. Due to particular simplicity of the EM algorithm, it has been widely applied in classification problems where the training data is annotated by low-cost noisy 'labelers' [JG03], [RYZ$^+$10]. In [WRW$^+$09] and [WBBP10], this EM approach has been applied to more complicated probabilistic models for image labeling tasks. However, the performance of these approaches are only empirically evaluated, and there is no analysis that proves performance guarantees. In particular, EM algorithms require an initial starting point which is typically randomly guessed. The algorithm is highly sensitive to this initialization, making it difficult to predict the quality of the resulting estimate.

Contributions. In this work, we provide a rigorous treatment of designing a crowdsourcing system with the aim of minimizing the budget to achieve completion of task with a certain reliability. We provide both an optimal graph construction (random regular bipartite graph) and an optimal algorithm for inference (low-rank approximation) on that graph. As the main result, we show that our algorithm performs as good as the best possible algorithm. The surprise lies in the fact that the optimality of our algorithm is established by comparing it with the best algorithm, one that is free to choose any graph, regular or irregular, and performs optimal estimation based on the information provided by an oracle about reliability of workers. In the process of establishing these results, we obtain a generalization of a celebrated result of Friedman, Kahn and Szemerédi [FKS89] on the spectrum of sparse random graphs.

Previous approaches focus on developing inference algorithms assuming that a graph is already given. None of the prior work on crowdsourcing provides any systematic treatment of the graph construction. We are the first to study both aspects of crowdsourcing together and, more importantly, establish optimality.

II. Main Results

In the crowdsourcing model introduced, we are interested in designing algorithms for two related problems: (i) how should the tasks be assigned to workers, i.e. the selection of the bipartite graph G; and (ii) given the responses from the workers, how should one estimate the correct answers. In what follows, we first address these two problems. For (i), we propose to utilize random regular bipartite graphs and for (ii), we propose to utilize certain low-rank matrix approximation based estimation procedure. We subsequently describe how we can integrate these two procedures into a single Budget-optimal Crowdsourcing system to achieve optimal performance.

A. Graph Generation

Assigning tasks to workers amounts to designing a bipartite graph G. Given m tasks to complete, the taskmaster first makes a choice of the left degree l (how many workers to assign to each task) and the right degree r (how many tasks to assign to each worker). The number of required workers n is then determined such that the total number of edges is consistent, that is $ml = nr$. To generate an (l, r)-regular bipartite graph we use a random graph generation scheme known as the *configuration model* in random graph literature [RU08], [Bol01]. In principle, one could use arbitrary bipartite graph G for task allocation. However, as we shall show in Section II-C, random regular graphs are sufficient to achieve order-optimal performance.

B. Inference Algorithm

Given G, let $\mathbf{A} = [\mathbf{A}_{ij}] \in \{0, 1, -1\}^{m\times n}$ denote the answers provided by the workers. In this section, we shall introduce a low-rank approximation based algorithm that takes $\mathbf{A}$ as input and produces an estimate for the unobserved solution vector $s = [s_i] \in \{-1, 1\}^m$. We then provide the performance guarantee of this algorithm. Surprisingly, as we will show in a later section, this simple inference algorithm can be used as a subroutine to achieve optimal performance.

```
Low-rank Approximation
Input: $\mathbf{A}$.
Output: Estimation $\hat{s}(\mathbf{A})$.
1:  Compute a pair of left and right singular vectors
    $(u, v)$ of $\mathbf{A}$ corresponding to the top singular value;
2:  If $\sum_{j:v_j\geq 0} v_j^2 < 1/2$, then output $\hat{s}(\mathbf{A}) = \text{sign}(-u)$;
3:  Otherwise, output $\hat{s}(\mathbf{A}) = \text{sign}(u)$;
```

In any case, the leading singular vector of $\mathbf{A}$ is not uniquely determined. Both (u, v) and $(-u, -v)$ are valid pairs of left and right singular vectors. To resolve this issue, our strategy is to choose the pair (u, v) if v has more 'mass' on the positive orthant than $-v$, that is $\sum_{j:v_j\geq 0} v_j^2 \geq \sum_{j:v_j<0} v_j^2$.

Let (u, v) be the pair of singular vectors after we resolve the ambiguity in the sign. The j-th entry of v represents our belief on how reliable worker j is, and our estimate is a weighted sum of the submitted answers weighted by workers' reliabilities:

$$\hat{s}_i \;=\; \text{sign}\Big(\sum_{j\in\partial i} \mathbf{A}_{ij} v_j\Big) \;, \qquad (2)$$

where $\partial i \subseteq [n]$ denotes the set of workers that are assigned task i. This follows from the fact that $u_i = \sum_{j\in\partial i} \mathbf{A}_{ij} v_j$. In Section III-A, we give in detail the intuition behind why the top left singular vector of $\mathbf{A}$ reveals the structure of the underlying unobserved answers.

With this estimate, we can show the following bound on the average number of error defined as the normalized Hamming distance:

$$d(s, \hat{s}) \;\;\equiv\;\; \frac{1}{m}\sum_{i=1}^{m} \mathbb{I}(s_i \neq \hat{s}_i) \;, \qquad (3)$$

where $\mathbb{I}(\cdot)$ denotes the indicator function. The proof of this result is given in Section III-A.

Theorem II.1. *For fixed l and r which are independent of m, assume that m tasks are assigned to $n = ml/r$ workers under the* spammer-hammer *model according to a random (l, r)-regular graph drawn from the configuration model. Then, for any $s \in \{\pm 1\}^m$, with probability $1 - m^{-\Omega(\sqrt{l})}$, the low-rank approximation algorithm achieves*

$$d(s, \hat{s}(\mathbf{A})) \;\;\leq\;\; \frac{C(\rho)}{lq} \;, \qquad (4)$$

where q is the probability that a randomly chosen worker is a hammer and $C(\rho)$ is a constant that only depends on $\rho \equiv l/r$.

Remarks about Theorem II.1. Now few remarks are in order. First, observe that this result is non-asymptotic and provides a concrete bound for any m. Second, the constant $C(\rho)$ depends continuously on ρ and is uniformly bounded over any closed interval $[\alpha, \beta]$ for $0 < \alpha \leq \beta < \infty$ (with bound dependent on α, β). To achieve optimal performance, we shall use the algorithm with $l = r = \Theta(1/q)$. Therefore, $\rho = 1$ and hence the constant $C(\rho) = C(1)$ can be treated as a universal constant independent of other problem parameters. Third, the choice of l (and r) depend on q to achieve average error less than $1/2$ (specifically, l must scale as $1/q$). Such an instance of algorithm (with $l = r = \Theta(1/q)$) will be used to design optimal crowdsourcing system. Thus, our system design requires (approximate) knowledge of q to achieve optimal performance to decide on the number of task replication, l. But beyond that, *neither* the graph selection *nor* the inference algorithm require the knowledge of q. A simple procedure is suggested that overcomes even this need of knowing q in Section II-D.

Run-time of Low-rank Approximation. We next show that $O(ml \log(m)/\log(lq))$ operations are sufficient to ensure that the performance guarantee in Theorem II.1 is achieved.

This is comparable to the simple majority voting which requires $\Omega(ml)$ operations.

As an implication of Lemma III.3, which is established as part of the proof of Theorem II.1, we obtain that when the bound (4) is non-trivial (i.e. $lq > C(\rho)$), there is a strict separation between the first singular value of $\mathbf{A}$ and the rest of the singular values. Precisely, $\sigma_2(\mathbf{A})/\sigma_1(\mathbf{A}) < C_1(\rho)/\sqrt{lq}$ with high probability, where $\sigma_i(\mathbf{A})$ is the i-th largest singular value of $\mathbf{A}$. This implies that the leading singular vector pair (u, v) is unique up to a sign.

We will be interested in the regime where $\mathbf{A}$ is sparse, that is $l, r = \Theta(1)$ and $\rho = \Theta(1)$. In this regime, the leading singular value and singular vectors of the matrix $\mathbf{A}$ can be computed efficiently, for instance, using power iteration [Ber92]. Each iteration requires $O(ml)$ operations. When the second singular value is less by a factor $C_1(\rho)/(lq)^{1/2} < 1$ compared to the first one, power iteration converges exponentially at rate determined by this factor. Therefore, for $\rho = \Theta(1)$ (in our optimal algorithm, $\rho = 1$), the Low-rank approximation takes $O(\log(m)/\log(lq))$ iterations to ensure the error bound mentioned. We summarize this in form the following Lemma which is proved in Section III-A.2.

Lemma II.2. *Under the hypothesis of Theorem II.1, the total* computational cost *required to achieve the bound in Theorem II.1 is $O(ml\log(m)/\log(lq))$.*

C. Optimality

As a taskmaster, the natural core optimization problem of our concern is how to achieve a certain reliability in our answers with minimum cost. Since we pay equal amount for all the task assignments, the cost is proportional to the total number of edges of the graph G. Here we compute the total budget sufficient to achieve a target error rate and show that this is within a constant factor from the necessary budget to achieve the given target error rate using any possible graph and any possible inference algorithm. The order-optimality is established with respect to all algorithms that operate in *one-shot*, i.e. all task assignments are done simultaneously, then an estimation is performed after all the answers are obtained.

Formally, consider a scenario where there are m tasks to complete and a target accuracy $\epsilon \in (0, 1/2)$. To measure accuracy, we use the average probability of error per task. We will show that $\Omega\big((1/q)\log(1/\epsilon)\big)$ assignments per task is necessary to achieve the target error rate:

$$\frac{1}{m}\sum_{i\in[m]} \mathbb{P}(s_i \neq \hat{s}_i) \leq \epsilon \, .$$

To design a system that can achieve the target error rate using a number of assignments close to this fundamental limit, we utilize the vanilla version of the graph assignment and inference algorithm described thus far to design an budget-optimal crowdsourcing system. The main idea is to obtain T independent estimates using T independent graphs and T independent sets of workers. Then, we can aggregate these independent estimates to achieve optimal performance.

```
Budget-optimal Crowdsourcing
Input: $m$ tasks, task degree $l$, number of groups $T$.
Output: estimate vector $\hat{s}$.
1:  For all $t \in [T]$ do
        Recruit $m$ workers;
        Generate an independent $(l,l)$-regular graph $G^{(t)}$
          from the configuration model;
        Assign $m$ tasks to $m$ workers according to $G^{(t)}$;
        Run the Low-rank Approximation algorithm;
        Set $\hat{s}^{(t)}$ as the estimation vector thus obtained;
2:  Output $\hat{s}_i = \mathrm{sign}(\sum_{t\in[T]} \hat{s}_i^{(t)})$.
```

Notice that this algorithm does not violate the one-shot scenario, since we can generate a single large graph $G = \cup_t G^{(t)}$ in advance. In particular, no information from one group is used in generating the graph for other group.

We state the following optimality result ($x \sim y$ indicates that x scales as y, i.e. $x = \Theta(y)$).

Theorem II.3. *Under the one-shot scenario, the following statements are true.*

(a) *Let Δ_{LB} the minimum cost per task* necessary *to achieve a target accuracy $\epsilon \in (0, 1/2)$ using any graph and any possible algorithm. Then*

$$\Delta_{\mathrm{LB}} \sim \frac{1}{q}\log\left(\frac{1}{\epsilon}\right) . \tag{5}$$

(b) *Let $\Delta_{\mathrm{Lowrank}}$ the minimum cost per task sufficient to achieve a target accuracy ϵ using the* Budget-optimal Crowdsourcing *system. Then there exists a universal constant M such that for all $m \geq M$,*

$$\Delta_{\mathrm{Lowrank}} \sim \frac{1}{q}\log\left(\frac{1}{\epsilon}\right) . \tag{6}$$

(c) *Let $\Delta_{\mathrm{Majority}}$ be the minimum cost per task necessary to achieve a target accuracy ϵ using the Majority voting scheme on any graph. Then*

$$\Delta_{\mathrm{Majority}} \sim \frac{1}{q^2}\log\left(\frac{1}{\epsilon}\right) . \tag{7}$$

The above (cf. (5) & (6)) establish the optimality of our algorithm. It is indeed surprising that regular graphs are sufficient to achieve this optimality. Further, the scaling of Majority voting is $(1/q^2)\log(1/\epsilon)$ which significantly worse than the optimal scaling of $(1/q)\log(1/\epsilon)$ of our algorithm. Finally, we emphasize that the low-rank approximation algorithm is quite efficient: it requires $O\big((1/q)\log(m)\log(1/\epsilon)\big)$ operations per task to achieve the target accuracy ϵ. It takes $O\big((1/q^2)\log(1/\epsilon)\big)$ operations for the simple majority voting to achieve the same reliability.

D. Discussions

It is worth pointing out some limitations and variations of our results, and interesting research directions:

Knowledge of q. We assumed the knowledge of q in selecting the degree $l = \Theta(1/q)$ in the design of the graph in the Budget-optimal Crowdsourcing system. Here is a simple way to overcome this limitation at the loss of only additional constant factor, i.e. scaling of cost per task still remains $\Theta(1/q \log(1/\epsilon))$. To that end, consider an incremental design in which at iteration k the system is designed assuming $q = 2^{-k}$ for $k \geq 1$. At iteration k, we design two replicas of the system as per the Budget-optimal Crowdsourcing for $q = 2^{-k}$. Now compare the estimates obtained by these two replicas for all m tasks. If they agree amongst $m(1 - 2\epsilon)$ tasks, then we stop and declare that as the final answer. Or else, we increase k to $k + 1$ and repeat. Note that by our optimality result, it follows that if 2^{-k} is less than the actual q then the iteration must stop with high probability. Therefore, the total cost paid is $\Theta(1/q \log(1/\epsilon))$ with high probability. Thus, even lack of knowledge of q does not affect the optimality of our algorithm.

More general models. For simplicity we assumed the spammer-hammer model, where $p_j \in \{1/2, 1\}$. A natural generalization of this is to allow any $p_j \in [0, 1]$. We expect that our algorithm and analysis can be strengthened to prove a bound on the error rate in this more general case.

An underlying assumption in our model is that the error probability of a worker does not depend on the particular task and all the tasks share an equal level of difficulty. Another underlying assumption in our model is that the workers are unbiased: the error probability of a worker does not depend whether the correct answer is a $+1$ or a -1. There is a more general model investigated in [WRW+09], which relaxes both of these assumptions. In formula, a worker j's response to a binary task i can be modeled as

$$\mathbf{A}_{ij} = \text{sign}(\mathbf{Z}_{i,j}) ,$$

where $\mathbf{Z}_{i,j} \sim \mathcal{N}(s_i + \alpha_j, r_i + \beta_j)$. Here, $s_i \in \{+1, -1\}$ represents the correct binary answer of task i, r_i represents the level of difficulty of task i, α_j represents the bias of worker j, and β_j represents the reliability of worker j. Most of the crowdsourcing models introduced so far can be reduced to a special case of this model. For example, the first crowdsourcing model introduced by Dawid and Skene [DS79] is equivalent to the above Gaussian model with $r_i = 0$ for all $i \in [m]$. The model we study in this paper is equivalent to the above Gaussian model with $\alpha_j = 0$ for all $j \in [n]$ and $r_i = 0$ for all $i \in [m]$. It is desirable to characterize how our algorithm works under this more general model.

III. Intuition and proofs

In this section, we present intuitions behind the low-rank approximation algorithm and provide a proof of Theorem II.1. We then provide a proof of optimality of our algorithm.

A. *Low-rank approximation*

A low-rank model is often used to capture the important aspects of datasets in matrix form. The data analysis technique using low-rank approximations of data matrices is referred to as principal component analysis (PCA) [Jol86]. PCA often reveals hidden structures in the data and has been successfully applied to applications including latent semantic indexing and spectral clustering.

A rank-1 approximation of our data matrix A can be easily computed using singular value decomposition (SVD). Let the singular value decomposition of A be

$$A \;=\; \sum_{i=1}^{\min\{m,n\}} u_i \sigma_i v_i^T ,$$

where $u_i \in \mathbb{R}^m$ and $v_i \in \mathbb{R}^n$ are the i-th left and right singular vectors, and $\sigma_i \in \mathbb{R}$ is the i-th singular value. Here and after, $(\cdot)^T$ denotes the transpose of a matrix or a vector. For simplicity, we use $u = u_1$ for the first left singular vector and $v = v_1$ for the first right singular vector. Singular values are typically assumed to be sorted in a non-increasing order satisfying $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$. Then, the optimal rank-1 approximation is given by a rank-1 projector $\mathcal{P}_1(\cdot) : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ such that

$$\mathcal{P}_1(A) \;=\; \sigma_1 u v^T , \tag{8}$$

It is a well known fact that $\mathcal{P}_1(A)$ minimizes the mean squared error. In formula,

$$\mathcal{P}_1(A) \;=\; \arg \min_{X:rank(X)\leq 1} \sum_{i,j} (A_{ij} - X_{ij})^2$$

In the problem of estimating the solutions of m tasks via crowdsourcing, it turns out that PCA provides good estimates. Consider an ideal case where G is a complete graph, and all the workers are hammers and provide the correct answers. Hence, there is no randomness in this example. Then, $A = s\mathbb{1}_n^T$, where $s \in \{\pm 1\}^m$ is the vector of correct solutions and $\mathbb{1}_n \in \mathbb{R}^n$ is the all ones vector. It is simple to see that A is a rank-1 matrix, since it is a multiplication of two 'rank-1' vectors. Therefore, u is equal to the correct solution s up to a scaling and sign (both $(1/\sqrt{m})s$ and $-(1/\sqrt{m})s$ are valid singular vectors).

Now, consider a case where all the workers are hammers but the graph is a (l, r)-regular graph. In this case, it is also true that u is equal to the correct solution s up to a scaling. Here is a sketch of the proof of this claim. Start with an all ones task vector $s = \mathbb{1}_m$. Then, $\mathbb{1}_m$ is an eigenvector of AA^T as $AA^T\mathbb{1}_m = lr\mathbb{1}_m$. It follows from the Perron-Frobenius theorem [HJ90] that $(1/\sqrt{m})\,\mathbb{1}_m$ is a left singular vector corresponding to the largest singular value of A. In other words, $u = (1/\sqrt{m})\,\mathbb{1}_m$.

Now, if s is not all ones vector, then $s = S\mathbb{1}_m$, where S is a diagonal matrix with $S_{ii} = s_i$. Note that S is also a orthogonal matrix such that $SS^T = S^TS = \mathbb{I}_m$, where $\mathbb{I}_m$ is the m dimensional identity matrix. Recall that for any orthogonal matrix Q, Qu is the first left singular vector of QA. Then, since we know that S is an orthogonal matrix and the first left singular vector of SA is $(1/\sqrt{m})\,\mathbb{1}_m$, it follows that the first singular vector of A is $u = (1/\sqrt{m})\,S^T\mathbb{1}_m = (1/\sqrt{m})\,s$. This proves the claim that u is equal to s up to a scaling. One important implication of the above argument is that the accuracy of this estimate does not depend on a particular choice of $s \in \{\pm 1\}^m$.

In the presence of spammers, the problem is more complicated since some workers can make errors. We get a random matrix $\mathbf{A}$ where the randomness comes from the construction of the graph G and the submitted answers on the edges of the graph. On expectation, a low-rank approximation of the random matrix $\mathbf{A}$ provides a good estimate. Recall that $p = [p_j] \in \{1/2, 1\}^n$. Since $\mathbb{E}[\mathbf{A}_{ij}] = (l/n)s_i(2p_j - 1)$, the expectation $\mathbb{E}[\mathbf{A}] = (l/n)s(2p - \mathbb{1}_n)^T$ is a rank-1 matrix whose first left singular vector is equal to s up to a scaling. We can consider the random matrix $\mathbf{A}$ as a perturbation of a rank-1 matrix: $\mathbf{A} = \mathbb{E}[\mathbf{A}] + \mathbf{Z}$, where the perturbation $\mathbf{Z}$ will be small in an appropriate sense if the number of spammers are small and the degree l is large. Building on this intuition, next we make this statement rigorous and provide a proof of the performance guarantee for the Low-rank Approximation algorithm.

1) Proof of Theorem II.1: Let $u = (u_1, \dots, u_m)^T$ be a left singular vector corresponding to the leading singular value of A. Ideally, we want to track each entry u_i for most realizations of the random matrix $\mathbf{A}$, which is difficult. Instead, we track the Euclidean distance between two vectors: $\|\frac{1}{\sqrt{m}}s - u\|$. The following lemma upper bounds the Hamming distance by the Euclidean distance.

Lemma III.1. *For any $s \in \{\pm 1\}^m$ and the Hamming distance $d(\cdot)$ defined in* (3)*,*

$$d(s, \mathrm{sign}(u)) \le \left\| \frac{1}{\sqrt{m}} s - u \right\|^2 . \tag{9}$$

Proof. The above lemma follows from a series of inequities:

$$\begin{aligned} \frac{1}{m}\sum_i \mathbb{I}(s_i \neq \mathrm{sign}(u_i)) &\le \frac{1}{m}\sum_i \mathbb{I}(s_i u_i \le 0) \\ &\le \frac{1}{m}\sum_i (s_i - \sqrt{m}u_i)^2 . \end{aligned}$$

□

To upper bound the Euclidean distance, we apply the next proposition to two rank-1 matrices: $\mathcal{P}_1(\mathbf{A})$ and $\mathbb{E}[\mathbf{A}|\mathbf{p}]$. For the proof of this proposition, we refer to the proof of a more general statement for general low-rank matrices in [KMO10, Remark 6.3].

Proposition III.2. *For two rank-1 matrices with singular value decomposition $M = x\sigma y^T$ and $M' = x'\sigma'(y')^T$, we have $\min\{\|x+x'\|, \|x-x'\|\} \le (\sqrt{2}/\sigma)\|M - M'\|_F$, where $\|x\| = \sqrt{\sum_i x_i^2}$ denotes the Euclidean norm and $\|X\|_F = \sqrt{\sum_{i,j}(X_{ij})^2}$ denotes the Frobenius norm. By symmetry this bound also holds with σ' in the denominator.*

For a given (random) quality vector $\mathbf{p} = (\mathbf{p}_1, \dots, \mathbf{p}_n)^T$ and any solution vector $s \in \{\pm 1\}^m$, $\mathbb{E}[\mathbf{A}|\mathbf{p}] = (l/m)s(2\mathbf{p} - \mathbb{1}_n)^T$ is a rank-1 matrix. Here, $\mathbb{1}_n$ denotes the n-dimensional all-ones vector. This matrix has a left singular vector $(1/\sqrt{m})s$ and a singular value $\sigma = \sqrt{lr/n}\|2\mathbf{p} - \mathbb{1}_n\|$. Recall that $\mathbf{p}_j = 1$ with probability q and $1/2$ with probability $1-q$, and $\mathbf{p}_j$'s are independent of one another. Since σ^2 is a sum of i.i.d. bounded random variables, we can applying Hoeffding's inequality to show that $\sigma \ge \sqrt{lrq/2}$ with probability $1 - e^{-\Omega(nq^2)}$.

$\mathcal{P}_1(\mathbf{A})$ is a rank-1 projection defined in (8). Notice that we have two choices for the left singular vector. Both u and $-u$ are valid singular vectors of $\mathcal{P}_1(\mathbf{A})$ and we do not know a priori which one is closer to $(1/\sqrt{m})s$. For now, let us assume that u is the one closer to the correct solution, such that $\|(1/\sqrt{m})s - u\| \le \|(1/\sqrt{m})s + u\|$. Later in this section, we will explain how we can identify u with high probability of success.

Applying Proposition III.2 to $\mathcal{P}_1(\mathbf{A})$ and $\mathbb{E}[\mathbf{A}|\mathbf{p}]$, we get that, with high probability,

$$\left\| \frac{1}{\sqrt{m}} s - u \right\| \le \frac{2}{\sqrt{lrq}} \big\| \mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A}) \big\|_F . \tag{10}$$

For any matrix X of rank-2, $\|X\|_F \le \sqrt{2}\|X\|_2$, where $\|X\|_2 \equiv \max_{\|x\|,\|y\| \le 1} x^T X y$ denotes the operator norm. Therefore, by triangular inequity,

$$\begin{aligned} &\big\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A})\big\|_F \\ &\quad \le \sqrt{2}\big\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A})\big\|_2 \\ &\quad \le \sqrt{2}\big\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathbf{A}\big\|_2 + \sqrt{2}\big\|\mathbf{A} - \mathcal{P}_1(\mathbf{A})\big\|_2 \\ &\quad \le 2\sqrt{2}\big\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathbf{A}\big\|_2 , \end{aligned} \tag{11}$$

where in the last inequity we used the fact that $\mathcal{P}_1(\mathbf{A})$ is the minimizer of $\|\mathbf{A} - X\|_2$ among all matrices X of rank one, whence $\|\mathbf{A} - \mathcal{P}_1(\mathbf{A})\|_2 \le \|\mathbf{A} - \mathbb{E}[\mathbf{A}|\mathbf{p}]\|_2$.

The following key technical lemma provides a bound on the operator norm of the difference between random matrix $\mathbf{A}$ and its (conditional) expectation. This lemma generalizes a celebrated bound on the second largest eigenvalue of d-regular random graphs by Friedman-Kahn-Szemerédi [FKS89], [FO05], [KMO10]. The proofs of this lemma is skipped here due to space limitations.

Lemma III.3. *Assume that an (l, r)-regular random bipartite graph G with m left nodes and $n = \Theta(m)$ right nodes is generated according to the configuration model. $\mathbf{A}$ is the weighted adjacency matrix of G with random weight $\mathbf{A}_{ij}$ assigned to each edge $(i, j) \in E$. With probability $1 - m^{-\Omega(\sqrt{l})}$,*

$$\|\mathbf{A} - \mathbb{E}[\mathbf{A}]\|_2 \;\le\; C'(\rho)A_{\max}(lr)^{1/4} , \tag{12}$$

where $|\mathbf{A}_{ij}| \le A_{\max}$ almost surely and $C'(\rho)$ is a constant that only depends on $\rho \equiv m/n$.

Under our model, $A_{\max} = 1$ since $\mathbf{A}_{ij} \in \{\pm 1\}$. We then apply this lemma to each realization of $\mathbf{p}$ and substitute this bound in (11). Together with (10) and (9), this finishes the proof Theorem II.1.

Now, we are left to prove that between u and $-u$ we can choose the one closer to $(1/\sqrt{m})s$. Given a rank-1 matrix $\mathcal{P}_1(\mathbf{A})$, there are two possible pairs of left and right 'normalized' singular vectors: (u, v) and $(-u, -v)$. Let $\mathcal{P}_+(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^n$ denote the projection onto the positive orthant such that $\mathcal{P}_+(v)_i = \mathbb{I}(v_i \ge 0)v_i$. Our strategy is to choose u to be our estimate if $\|\mathcal{P}_+(v)\|^2 \ge 1/2$ (and

$-u$ otherwise). We claim that with high probability the pair (u, v) chosen according to our strategy satisfies

$$\|(1/\sqrt{m})s - u\| \le \|(1/\sqrt{m})s + u\| \; . \tag{13}$$

Assume that the pair (u, v) is the one satisfying the above inequality. Denote the singular vectors of $\mathbb{E}[\mathbf{A}|\mathbf{p}]$ by $x = (1/\sqrt{m})s$ and $y = (1/\|2\mathbf{p} - \mathbb{1}_n\|)(2\mathbf{p} - \mathbb{1}_n)$, and singular value $\sigma' = \|\mathbb{E}[\mathbf{A}|\mathbf{p}]\|_2$. Let $\sigma = \|\mathcal{P}_1(\mathbf{A})\|_2$. Then, by Proposition III.2 and the triangular inequality,

$$\begin{aligned}\|y - v\| &= \Big\|\frac{1}{\sigma'}\mathbb{E}[\mathbf{A}|\mathbf{p}]^T x - \frac{1}{\sigma}\mathcal{P}_1(\mathbf{A})^T u\Big\| \\ &\le \Big\|\frac{1}{\sigma'}\mathbb{E}[\mathbf{A}|\mathbf{p}]^T(x-u)\Big\| + \Big\|\frac{1}{\sigma'}(\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A}))^T u\Big\| \\ &\quad + \Big\|\Big(\frac{1}{\sigma'} - \frac{1}{\sigma}\Big)\mathcal{P}_1(\mathbf{A})^T u\Big\| \\ &\le \frac{C_1}{(lrq^2)^{1/4}} \; .\end{aligned}$$

The first term is upper bounded by $\|(1/\sigma')\mathbb{E}[\mathbf{A}|\mathbf{p}]^T(x - u)\| \le \|x - u\|$, which is again upper bounded by $C_2/(lrq^2)^{1/4}$ using (10). The second term is upper bounded by $\|(1/\sigma')(\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A}))^T u\| \le (1/\sigma')\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A})\|_2$, which is again upper bounded by $C_3/(lrq^2)^{1/4}$ using (12) and $\sigma' \ge (1/2)\sqrt{lrq}$. The third term is upper bounded by $\|\Big(\frac{1}{\sigma'} - \frac{1}{\sigma}\Big)\mathcal{P}_1(\mathbf{A})^T u\| \le |\sigma - \sigma'|/\sigma'$, which is again upper bounded by $C_4/(lrq^2)^{1/4}$ using triangular inequality: $(1/\sigma')\big|\|\mathbb{E}[\mathbf{A}|\mathbf{p}]\|_2 - \|\mathcal{P}_1(\mathbf{A})\|_2\big| \le (1/\sigma')\|\mathbb{E}[\mathbf{A}|\mathbf{p}] - \mathcal{P}_1(\mathbf{A})\|_2$.

This implies that

$$\begin{aligned}\|\mathcal{P}_+(v)\| &\ge \|y\| - \|y - \mathcal{P}_+(v)\| \\ &\ge 1 - \|y - v\| \\ &\ge 1 - C_1/(lrq^2)^{1/4} \; ,\end{aligned}$$

where the second inequality follows from the fact that $\mathbf{p}_j \ge 1/2$ which implies that all entries of y are non-negative. Notice that we can increase the constant $C(\rho)$ in the bound (4) of the main theorem such that we only need to restrict our attention to $(lrq^2)^{1/4} > 4C_1$. This proves that the pair (u, v) chosen according to our strategy satisfy (13), which is all we need in order to prove Theorem II.1.

2) Proof of Lemma II.2: Power iteration is a simple procedure for computing the singular vector of a matrix A corresponding to the largest singular value. Power iteration is especially efficient when A sparse, since it iterates by applying the matrix twice at each iteration. We denote the i-th largest singular value by $\sigma_i(A)$, such that $\sigma_1(A) \ge \sigma_2(A) \ge \ldots \ge 0$. Let u be the left singular vector corresponding to the largest singular value. Then, our estimate of u after k iterations is

$$\begin{aligned}\widetilde{x}^{(k)} &= AA^T x^{(k-1)} \; , \\ x^{(k)} &= \frac{1}{\|\widetilde{x}^{(k)}\|}\widetilde{x}^{(k)} \; ,\end{aligned}$$

with a random initialization $x^{(0)}$. The convergence of the sequence $x^{(k)}$ to u is not sensitive to a particular initialization, and we can initialize each entry of $x^{(0)}$, for example, as i.i.d. Gaussian random variable.

Let $\mathcal{P}_u(x) = (u^T x)u$ denote the projection onto the subspace spanned by u, and $\mathcal{P}_{u^\perp}(x) = x - (u^T x)u$ be the projection onto the complement subspace. Then, by singular value decomposition of A, it follows that $\mathcal{P}_u(\widetilde{x}^{(k)}) = (\sigma_1(A))^2\mathcal{P}_u(x^{(k-1)})$ and $\mathcal{P}_{u^\perp}(\widetilde{x}^{(k)}) \le (\sigma_2(A))^2\mathcal{P}_{u^\perp}(x^{(k-1)})$. Then,

$$\frac{\|\mathcal{P}_{u^\perp}(x^{(k)})\|}{\|\mathcal{P}_u(x^{(k)})\|} \le \Big(\frac{\sigma_2(A)}{\sigma_1(A)}\Big)^2\frac{\|\mathcal{P}_{u^\perp}(x^{(k-1)})\|}{\|\mathcal{P}_u(x^{(k-1)})\|} \; . \tag{14}$$

Since $\|\mathcal{P}_{u^\perp}(x^{(k)})\|$ is the error in our estimate, this implies that the error in our estimation of the vector u decays exponentially given that $\sigma_2(A)$ is strictly smaller than $\sigma_1(A)$. Under the crowdsourcing model, we can even show a stronger result than just a strict inequality. Namely,

$$\frac{\sigma_2(\mathbf{A})}{\sigma_1(\mathbf{A})} \le \frac{C_1}{(lrq^2)^{1/4}} \; , \tag{15}$$

with high probability. From the proof of Theorem II.1, recall that $\mathbb{E}[\mathbf{A}]$ is a rank-1 matrix with $\|\mathbb{E}[\mathbf{A}]\|_2 \ge (1/2)\sqrt{lrq}$ with high probability. By Lemma III.3, we know that $\|\mathbf{A} - \mathbb{E}[\mathbf{A}]\|_2 \le C_2(lr)^{1/4}$. Now applying Weyl's inequality [HJ90, Theorem 4.3.1], it immediately follows from that $\sigma_1(\mathbf{A}) \ge (1/2)\sqrt{lrq} - C_2(lr)^{1/4} \ge C_3\sqrt{lrq}$ and $\sigma_2(\mathbf{A}) \le C_2(lr)^{1/4}$. Here we used the fact that in the regime where Theorem II.1 is non-trivial, we can assume that $\sqrt{lrq^2} \ge C(\rho)$. Weyl's inequality applied to non-symmetric matrices states that for two matrices M and M' of the same dimensions, $\sigma_k(M) - \sigma_1(M') \le \sigma_k(M + M') \le \sigma_k(M) + \sigma_1(M')$.

Substituting (15) in (14), we get that

$$\frac{\|\mathcal{P}_{u^\perp}(x^{(k)})\|}{\|\mathcal{P}_u(x^{(k)})\|} \le \Big(\frac{C_1}{(lrq^2)^{1/4}}\Big)^{2k} 2\sqrt{m} \; , \tag{16}$$

with high probability. Here we used the concentration of measure result on Gaussian random variables: $(\|\mathcal{P}_{u^\perp}(x^{(0)})\|)/(\|\mathcal{P}_u(x^{(0)})\|) \le 2\sqrt{m}$ with high probability.

Now, for the error bound in Theorem II.1 to hold after a finite k iterations, all we need is for our estimate $x^{(k)}$ to satisfy, $\|x^{(k)} - (1/\|s\|)s\| \le C_5/(lrq^2)^{1/4}$. By triangular inequality,

$$\|x^{(k)} - (1/\|s\|)s\| \le \|(1/\|s\|)s - u\| + \|x^{(k)} - u\| \; .$$

We already proved in Section III-A.1 that the first term is bounded with appropriate bound. For the second term, notice that $\|x^{(k)} - u\|^2 = 2\|\mathcal{P}_{u^\perp}(x^{(k)})\|$. By (16), it follows that $k = O(\log(m)/\log(lrq^2))$ is sufficient to guarantee that the error bound in Theorem II.1 holds with high probability.

B. Optimality under One-Shot: Proof of Theorem II.3

In order to compute the fundamental limit on cost-accuracy tradeoff, we need a lower bound on the achievable accuracy using any possible scheme. Let us consider an oracle estimator that makes the optimal decision based on information provided by an oracle. Further, assume that the oracle gives information on which workers are hammers and which are spammers. This estimator only makes mistakes on tasks whose neighbors are all spammers, in which case the estimator flips a fair coin to make a decision. If we denote the degree of node i by l_i, the error rate $\mathbb{P}(\hat{s}_i \neq s_i)$ is $(1/2)(1-q)^{l_i}$. Note that no algorithm with only information on $\{A_{ij}\}$ can produce more accurate estimate than the oracle estimator. Therefore, for any estimate $\hat{s}$ that is a function of $\{A_{ij}\}_{(i,j)\in E}$, we have the following minimax bound.

$$\begin{aligned}\inf_{\hat{s}} \sup_{s\in\{\pm 1\}^m} \mathbb{P}(s \neq \hat{s}) &\geq \frac{1}{m}\sum_{i\in[m]} \frac{1}{2}(1-q)^{l_i} \\ &\geq \frac{1}{2}(1-q)^{|E|/m} ,\end{aligned}$$

where the second inequality follows by convexity and $|E| = \sum_i l_i$ is the total number of edges. This implies that in order to achieve a target accuracy ϵ using any possible algorithm, it is necessary to have $|E| \sim (m/q)\log(1/\epsilon)$. This gives $\Delta_{\rm LB} \sim (1/q)\log(1/\epsilon)$. Let us emphasize that the necessary condition on the budget in (5) is completely general and holds for any graph, regular or irregular.

Next, consider the proposed Budget-optimal Crowdsourcing algorithm. To achieve the optimal performance, we run T independent estimation procedure to get T independent estimates of each s_i. Let $\hat{s}_i^{(t)}$ denote the estimate of s_i produces by the t-th group of workers for $t \in [T]$. Set $l \sim 1/q$ with large enough constant such that $(1/m)\sum_{i\in[m]} \mathbb{I}(s_i \neq \hat{s}_i^{t)}) \leq 1/8$ with probability $1 - m^{-\Omega(\sqrt{l})}$ by Theorem II.1. By symmetry of the graph selection procedure, the estimate accuracy is invariant of a particular choice of task i. Then, $\mathbb{P}(s_i \neq \hat{s}_i^{(t)}) \leq 1/8 + m^{-\Omega(\sqrt{l})}$. Then for $m \geq e^{\Omega(1/\sqrt{l})}$, $\mathbb{P}(s_i \neq \hat{s}_i^{(t)}) \leq 1/4$. Note that $e^{\Omega(1/\sqrt{l})} = O(1)$. Further, $\{\hat{s}_i^{(t)}\}_{t\in[T]}$ are independent given s_i.

Our final estimate after T runs is $\hat{s}_i = \mathrm{sign}\Big(\sum_{t=1}^T \hat{s}_i^{(t)}\Big)$. By concentration of measure result, the error rate is upper bounded by $\mathbb{P}(\hat{s}_i \neq s_i) \leq e^{-(1/8)T}$. To achieve accuracy ϵ we need to have $T \sim \log(1/\epsilon)$. The minimum cost per task *sufficient* to achieve a target accuracy ϵ scales as lT, which gives $\Delta_{\rm Lowrank} \sim (1/q)\log\big(1/\epsilon\big)$.

Now, consider a naive majority voting algorithm. Majority voting simply follows what the majority of workers agree on. In formula, $\hat{s}_i = \mathrm{sign}(\sum_{j\in\partial i} A_{ij})$, where ∂i denotes the neighborhood of node i in the graph. It makes a random choice when there is a tie. When we have many spammers in the crowd, majority voting is prone to make mistakes since it gives the same weight to both the estimates provided by spammers and those of hammers. This limitation is captured in the following lower bound.

Lemma III.4. *There exists a numerical constant C_2 such that the error rate achieved using majority voting scheme is lower bounded by $\mathbb{P}(\hat{s}_i \neq s_i) \geq e^{-C_2(l_i q^2+1)}$, where l_i is the degree of node i.*

The proof of this Lemma is skipped due to space limitations. Now from convexity it follows that

$$\frac{1}{m}\sum_{i\in[m]} \mathbb{P}(s_i \neq \hat{s}_i) \geq e^{-C_2((1/m)|E|q^2+1)} .$$

Then, with majority voting scheme, the minimum cost per task *necessary* to achieve a target accuracy ϵ is $\Delta_{\rm Majority} \sim (1/q^2)\log\big(1/\epsilon\big)$.

References

[Ber92] M. W. Berry, *Large scale sparse singular value computations*, International Journal of Supercomputer Applications **6** (1992), 13–49.

[Bol01] B. Bollobás, *Random Graphs*, Cambridge University Press, January 2001.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin, *Maximum likelihood from incomplete data via the em algorithm*, Journal of the Royal Statistical Society. Series B (Methodological) **39** (1977), no. 1, pp. 1–38.

[DS79] A. P. Dawid and A. M. Skene, *Maximum likelihood estimation of observer error-rates using the em algorithm*, Journal of the Royal Statistical Society. Series C (Applied Statistics) **28** (1979), no. 1, 20–28.

[FKS89] J. Friedman, J. Kahn, and E. Szemerédi, *On the second eigenvalue in random regular graphs*, Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing (Seattle, Washington, USA), ACM, may 1989, pp. 587–598.

[FO05] U. Feige and E. Ofek, *Spectral techniques applied to sparse random graphs*, Random Struct. Algorithms **27** (2005), no. 2, 251–275.

[HJ90] R. A. Horn and C. R. Johnson, *Matrix analysis*, Cambridge University Press, 1990.

[JG03] R. Jin and Z. Ghahramani, *Learning with multiple labels*, Advances in neural information processing systems (2003), 921–928.

[Jol86] I. T. Jolliffe, *Principal component analysis*, Springer-Verlag, 1986.

[KMO10] R. H. Keshavan, A. Montanari, and S. Oh, *Matrix completion from a few entries*, IEEE Trans. Inform. Theory **56** (2010), no. 6, 2980–2998.

[RU08] T. Richardson and R. Urbanke, *Modern Coding Theory*, Cambridge University Press, march 2008.

[RYZ$^+$10] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, *Learning from crowds*, J. Mach. Learn. Res. **99** (2010), 1297–1322.

[WBBP10] P. Welinder, S. Branson, S. Belongie, and P. Perona, *The Multidimensional Wisdom of Crowds*, 2424–2432.

[WRW$^+$09] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan, *Whose vote should count more: Optimal integration of labels from labelers of unknown expertise*, Advances in Neural Information Processing Systems **22** (2009), 2035–2043.