Semisupervised Clustering, AND-Queries and Locally Encodable Source Coding

Arya Mazumdar
College of Information & Computer Sciences
University of Massachusetts Amherst
Amherst, MA 01003
arya@cs.umass.edu

Soumyabrata Pal
College of Information & Computer Sciences
University of Massachusetts Amherst
Amherst, MA 01003
soumyabratap@umass.edu

Abstract

Source coding is the canonical problem of data compression in information theory. In a *locally encodable* source coding, each compressed bit depends on only few bits of the input. In this paper, we show that a recently popular model of semisupervised clustering is equivalent to locally encodable source coding. In this model, the task is to perform multiclass labeling of unlabeled elements. At the beginning, we can ask in parallel a set of simple queries to an oracle who provides (possibly erroneous) binary answers to the queries. The queries cannot involve more than two (or a fixed constant number Δ of) elements. Now the labeling of all the elements (or clustering) must be performed based on the (noisy) query answers. The goal is to recover all the correct labelings while minimizing the number of such queries. The equivalence to locally encodable source codes leads us to find lower bounds on the number of queries required in variety of scenarios. We are also able to show fundamental limitations of pairwise 'same cluster' queries - and propose pairwise AND queries, that provably performs better in many situations.

1 Introduction

Suppose we have n elements, and the ith element has a label $X_i \in \{0, 1, \ldots, k-1\}, \forall i \in \{1, \ldots, n\}$. We consider the task of learning the labels of the elements (or learning the label vector). This can also be easily thought of as a clustering problem of n elements into k clusters, where there is a ground-truth clustering[1]. There exist various approaches to this problem in general. In many cases some similarity values between pair of elements are known (a high similarity value indicate that they are in the same cluster). Given these similarity values (or a weighted complete graph), the task is equivalent to to graph clustering; when perfect similarity values are known this is equivalent to finding the connected components of a graph.

A recent approach to clustering has been via crowdsourcing. Suppose there is an oracle (expert labelers, crowd workers) with whom we can make pairwise queries of the form "do elements u and v belong to the same cluster?". We will call this the 'same cluster' query (as per [3]). Based on the answers from the oracle, we then try to reconstruct the labeling or clustering. This idea has seen a recent surge of interest especially in the entity resolution research (see, for e.g. [34, 31, 7, 20]). Since each query to crowd workers cost time and money, a natural objective will be to minimize the number of queries to the oracle and still recover the clusters exactly. Carefully designed adaptive and interactive querying algorithms for clustering has also recently been developed [34, 31, 7, 22, 21]. In

[1]The difference between clustering and learning labels is that in the case of clustering it is not necessary to know the value of the label for a cluster. Therefore any unsupervised labeling algorithm will be a clustering algorithm, however the reverse is not true. In this paper we are mostly concerned about the labeling problem, hence our algorithms (upper bounds) are valid for clustering as well.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

particular, the query complexity for clustering with a k-means objective had recently been studied in [3], and there are significant works in designing optimal crowdsourcing schemes in general (see, [12, 13, 29, 35, 15]). Note that, a crowd worker may potentially handle more than two elements at a time; however it is of interest to keep the number of elements involved in a query as small as possible. As an example, recent work in [32] considers triangle queries (involving three elements in a query). Also crowd workers can compute some simple functions on this small set of inputs - instead of answering a 'same cluster' query. But again it is desirable that the answer the workers provide to be simple, such as a binary answer.

The queries to the oracle can be asked adaptively or non-adaptively. For the clustering problem, both the adaptive version and the nonadaptive versions have been studied. While both versions have obvious advantages and disadvantages, for crowdsourcing applications it is helpful to have a parallelizable querying scheme in most scenarios for faster response-rate and real time analysis. In this paper, we concentrate on the nonadaptive version of the problem, i.e., we perform the labeling algorithm after all the query answers are all obtained.

Budgeted crowdsourcing problems can be quite straight-forwardly viewed as a canonical source-coding or source-channel coding problem of information theory (e.g., see the recent paper [14]). A main contribution of our paper is to view this as a *locally encodable* source coding problem: a data compression problem where each compressed bit depends only on a constant number of input bits. The notion of locally encodable source coding is not well-studied even within information theory community, and the only place where it is mentioned to the best of our knowledge is in [24], although the focus of that paper is a related notion of *smooth* encoding. Another related notion of *local decoding* seem to be much more well-studied [19, 18, 16, 27, 5, 26, 4, 33].

By posing the querying problem as such we can get a number of information theoretic lower bounds on the number of queries required to recover the correct labeling. We also provide nonadaptive schemes that are near optimal. Another of our main contributions is to show that even within queries with binary answers, 'same cluster' queries (or XOR queries) may not be the best possible choice. A smaller number of queries can be achieved for approximate recovery by using what we call an AND query. Among our settings, we also consider the case when the oracle gives incorrect answers with some probability. A simple scheme to reduce errors in this case could be to take a majority vote after asking the same question to multiple different crowd workers. However, often that is not sufficient. Experiments on several real datasets (see [21]) with answers collected from Amazon Mechanical Turk [9, 30] show that majority voting could even increase the errors. Interestingly, such an observation has been made by a recent paper as well [28, Figure 1]. The probability of error of a query answer may also be thought of as the aggregated answer after repeating the query several times. Once the answer has been aggregated, it cannot change – and thus it reduces to the model where repeating the same question multiple times is not allowed. On the other hand, it is usually assumed that the answers to different queries are independently erroneous (see [10]). Therefore we consider the case where repetition of a same query multiple times is not allowed[2], however different queries can result in erroneous answers independently.

In this case, the best known algorithms need $O(n \log n)$ queries to perform the clustering with two clusters [21]. We show that by employing our AND querying method $(1-\delta)$-proportion of all labels in the label vector will be recovered with only $O(n \log \frac{1}{\delta})$ queries.

Along the way, we also provide new information theoretic results on fundamental limits of locally encodable source coding. While the the related notion of locally decodable source code [19, 16, 27, 5], as well as smooth compression [24, 27] have been studied, there was no nontrivial result known related to locally encodable codes in general. Although the focus of this paper is primarily theoretical, we also perform a real crowdsourcing experiment to validate our algorithm.

2 Problem Setup and Information Theoretic View

For n elements, consider a label vector $\boldsymbol{X} \in \{0, \ldots, k-1\}^n$, where X_i, the ith entry of $\boldsymbol{X}$, is the label of the ith element and can take one of k possible values. Suppose $P(X_i = j) = p_j \forall j$ and X_i's are independent. In other words, the prior distribution of the labels is given by the vector

[2]Independent repetition of queries is also theoretically not interesting, as by repeating any query just $O(\log n)$ times one can reduce the probability of error to near zero.

$\boldsymbol{p} \equiv (p_0, \ldots, p_{k-1})$. For the special case of $k = 2$, we denote $p_0 \equiv 1 - p$ and $p_1 \equiv p$. While we emphasize on the case of $k = 2$ our results extends in the case of larger k, as will be mentioned.

A query $Q : \{0, \ldots, k-1\}^{\Delta} \to \{0, 1\}$ is a deterministic function that takes as argument Δ labels, $\Delta \ll n$, and outputs a binary answer. While the query answer need not be binary, we restrict ourselves mostly to this case for being the most practical choice.

Suppose a total of m queries are made and the query answers are given by $\boldsymbol{Y} \in \{0, 1\}^m$. The objective is to reconstruct the label vector $\boldsymbol{X}$ from $\boldsymbol{Y}$, such that the number of queries m is minimized.

We assume our recovery algorithms to have the knowledge of $\boldsymbol{p}$. This prior distribution, or the relative sizes of clusters, is usually easy to estimate by subsampling a small ($O(\log n)$) subset of elements and performing a complete clustering within that set (by, say, all pairwise queries). In many prior works, especially in the recovery algorithms of popular statistical models such as stochastic block model, it is assumed that the relative sizes of the clusters are known (see [1]).

We also consider the setting where query answers may be erroneous with some probability of error. For crowdsourcing applications, this is a valid assumption since many times even expert labelers can make errors, and such assumption can be made. To model this we assume each entry of $\boldsymbol{Y}$ is flipped independently with some probability q. Such independence assumption has been used many times previously to model errors in crowdsourcing systems (see, e.g., [10]). While this may not be the perfect model, we *do not allow a single query to be repeated multiple times in our algorithms* (see the Introduction for a justification). For the analysis of our algorithm we just need to assume that the answers to different queries are independent. While we analyze our algorithms under these assumptions for theoretical guarantees, it turns out that even in real crowdsourcing situations our algorithmic results mostly follow the theoretical results, giving further validation to the model.

For the $k = 2$ case, and when $q = 0$ (perfect oracle), it is easy to see that n queries are sufficient for the task. One simply compares every element with the first element. This does not extend to the case when $k > 2$: for perfect recovery, and without any prior, one must make $O(n^2)$ queries in this case. When $q > 0$ (erroneous oracle), it has been shown that a total number of $O(\gamma n k \log n)$ queries are sufficient [21], where γ is the ratio of the sizes of the largest and smallest clusters.

Information theoretic view. The problem of learning a label vector $\boldsymbol{X}$ from queries is very similar to the canonical source coding (data compression) problem from information theory. In the source coding problem, a (possibly random) vector $\boldsymbol{X}$ is 'encoded' into a usually smaller length binary vector called the *compressed vector*[3] $\boldsymbol{Y} \in \{0, 1\}^m$. The decoding task is to again obtain $\boldsymbol{X}$ from the compressed vector $\boldsymbol{Y}$. It is known that if each entry of $\boldsymbol{X}$ is independently distributed according to $\boldsymbol{p}$, then $m \approx nH(\boldsymbol{p})$ is both necessary and sufficient to recover $\boldsymbol{x}$ with high probability, where $H(\boldsymbol{p}) = -\sum_i p_i \log p_i$ is the entropy of $\boldsymbol{p}$.

We can cast our problem in this setting naturally, where entries of $\boldsymbol{Y}$ are just answers to queries made on $\boldsymbol{X}$. The main difference is that in source coding each Y_i may potentially depend on all the entries of $\boldsymbol{X}$; while in the case of label learning, each Y_i may depend on only Δ of the X_is.

We call this *locally encodable source coding*. This terminology is analogous to the recently developed literature on locally decodable source coding [19, 16]. It is called locally encodable, because each compressed bit depend only on Δ of the source (input) bits. For locally decodable source coding, each bit of the reconstructed sequence $\hat{\boldsymbol{X}}$ depends on at most a prescribed constant number Δ of bits from the compressed sequence. Another closely related notion is that of 'smooth compression', where each source bit contributes to at most Δ compressed bits [24]. Indeed, in [24], the notion of locally encodable source coding is also present where it was called robust compression. We provide new information theoretic lower bounds on the number of queries required to reconstruct $\boldsymbol{X}$ exactly and approximately for our problem.

For the case when there are only two labels, the 'same cluster' query is equivalent to an Boolean XOR operation between the labels. There are some inherent limitations to these functions that prohibit the 'same cluster' queries to achieve the best possible number of queries for the 'approximate' recovery of labeling problem. We use an old result by Massey [17] to establish this limitation. We show that, instead using an operation like Boolean AND, much smaller number of queries are able to recover most of the labels correctly.

[3]The compressed vector is not necessarily binary, nor it is necessarily smaller length.

We also consider the case when the oracle gives faulty answer, or $\boldsymbol{Y}$ is corrupted by some noise (the *binary symmetric channel*). This setting is analogous to the problem of *joint source-channel coding*. However, just like before, each encoded bit must depend on at most Δ bits. We show that for the approximate recovery problem, AND queries are again performing substantially well. In a real crowdsourcing experiment, we have seen that if crowd-workers have been provided with the same set of pairs and being asked for 'same cluster' queries as well as AND queries, the error-rate of AND queries is lower. The reason is that for a correct 'no' answer in an AND query, a worker just need to know one of the labels in the pair. For a 'same cluster' query, both the labels must be known to the worker for any correct answer.

There are multiple reasons why one would ask for a 'combination' or function of multiple labels from a worker instead of just asking for a label itself (a 'label-query'). Note that, asking for labels will never let us recover the clusters in less than n queries, whereas, as we will see, the queries that combine labels will. Also in case of erroneous answer with AND queries or 'same cluster' queries, we have the option of not repeating a query, and still reduce errors. No such option is available with direct label-queries.

Remark 1. Note that, using only 'same cluster' queries at best the complete clustering can be recovered, and it is not possible to recover the labeling unless some other information is also available. Indeed, if the prior $\boldsymbol{p} = (p_0, \ldots, p_{k-1})$ is known, and $p_i \neq p_j \forall i, j$, then by looking at the complete clustering it is possible to figure out the labeling (or to correctly assign the clusters their respective labels, since the label vector must belong to the 'typical set' of $\boldsymbol{p}$. This implies, for the case of two clusters, if $p \neq \frac{1}{2}$ then the labeling can be inferred with high probability from the clustering.

Contributions. In summary our contributions can be listed as follows.

1. Noiseless queries and exact recovery (Sec. 3.1): For two labels/clusters, we provide a querying scheme that asks $n - \Theta(n/\log n)$ nonadaptive pairwise 'same cluster' queries, and recovers the labels with high probability, for a range of prior probabilities. We also show that, this result is order-wise optimal. If instead we involve $\Delta \geq 3$ elements in each of the queries, then with $\alpha n, \alpha < 1$ number of nonadaptive XOR queries we recover all labels with high probability, for a range of prior probabilities. We also provide a new lower bound that is strictly better than $nH(\boldsymbol{p})$ for some $\boldsymbol{p}$.

2. Noiseless queries and approximate recovery (Sec. 3.2): We provide a new lower bound on the number of queries required to recover $(1-\delta)$ fraction of the labels $\delta > 0$. We also show that 'same cluster' queries are insufficient, and propose a new querying strategy based on AND operation that performs substantially better.

3. Noisy queries and approximate recovery (Sec. 3.3). For this part we assumed the query answer to be k-ary ($k \geq 2$) where k is the number of clusters. This section contains the main algorithmic result that uses the AND queries as main primitive. We show that, even in the presence of noise in the query answers, it is possible to recover $(1-\delta)$ proportion of all labels correctly with only $O(n \log \frac{k}{\delta})$ nonadaptive queries. We validate this theoretical result in a real crowdsourcing experiment in Sec. 4.

3 Main results and techniques

3.1 Noiseless queries and exact recovery

In this scenario we assume the query answer from the oracle to be perfect and we wish to get back all of the original labels exactly without any error. Each query is allowed to take only Δ labels as input. When $\Delta = 2$, we are allowed to ask only pairwise queries. Let us consider the case when there are only two labels, i.e., $k = 2$. That means the labels $X_i \in \{0, 1\}, 1 \leq i \leq n$, are iid Bernoulli($p$) random variable. Therefore the number of queries m that are necessary and sufficient to recover all the labels with high probability is approximately $nH(p) \pm o(n)$ where $H(x) \equiv -x \log_2 x - (1-x) \log_2 (1-x)$ is the binary entropy function. However the sufficiency part here does not take into account that each query can involve only Δ labels.

3.1.1 Same cluster queries ($\Delta = 2$)

We warm up with the case of only two labels (clusters), and the same cluster queries. For two labels, a same cluster query simply amounts to being the modulo 2 sum of the two label values. It is easy to

see that querying the first label (X_1) with every other label allows us to infer the clustering of the labels since we can simply group the labels which are same as the first label and the labels which are different to the first label separately. As mentioned in Remark 1, if $p \neq \frac{1}{2}$, then just by counting the sizes of the groups, it is possible to get the labels correctly as well (among the two possible labelings, given the clustering). The query complexity in this case is $n-1$ but we can have the following scheme that uses less than $n-1$ queries by building on the aforementioned idea.

Querying scheme. Our scheme works in the following manner. First, we partition the n elements into d disjoint and equal sized groups each containing $\frac{n}{d}$ elements (assume $d|n$). For each group, we query the first element of the group with every other element of the group. Now, for each group we can cluster the labels and we will identify

- the smaller cluster with the label `1` and the larger cluster with the label `0`, when $p < \frac{1}{2}$
- the smaller cluster with the label `0` and the larger cluster with the label `1`, when $p > \frac{1}{2}$.

Going forward, let us just assume $p < \frac{1}{2}$ without any loss of generality. Finally we can aggregate all the elements identified with the label `1` and with label `0` and return them as our final output. The total number of queries required in this scheme is $d(\frac{n}{d} - 1) = n - d$.

Theorem 1. *For the querying scheme described above, $n - \frac{nD(\frac{1}{2}||p)}{2\log n}$ same cluster queries are sufficient to recover the label vector $\boldsymbol{X}$ with probability at least $1 - \frac{D(\frac{1}{2}||p)}{2n\log n}$ where $D(p||q) \equiv p\ln\frac{p}{q} + (1-p)\ln\frac{1-p}{1-q}$ is the Kullback-Leibler Divergence between two Bernoulli distributed random variables with parameters p and q respectively.*

Proof. Let us set $d = \frac{nD(\frac{1}{2}||p)}{2\log n}$. We omit the use of ceilings and floor to maintain clarity. Within a group of elements, we are able to obtain the complete clustering. We will fail to obtain the labeling only if the larger cluster has the true label `1`. However this happens with probability at most $2^{-\frac{nD(\frac{1}{2}||p)}{d}}$ [6]. Since there are d groups, the probability that we fail to recover the labels in any one of the groups is at most $d2^{-\frac{nD(\frac{1}{2}||p)}{d}} = \frac{d}{n^2} = \frac{D(\frac{1}{2}||p)}{2n\log n}$. The total number of queries is $n - d = n - \frac{nD(\frac{1}{2}||p)}{2\log n}$. □

Now, we will show a matching lower bound that proves that the reduction on query complexity presented in the scheme described above to be tight up to constant factors. In particular, we prove the following theorem.

Theorem 2. *Consider the binary labeling problem with pairwise queries. If the number of 'same cluster' queries is less than $n - \frac{(2+\epsilon)nD(\frac{1}{2}||p)}{\log n}$ for any positive constant $\epsilon > 0$, then any querying scheme will fail to recover the labels of all elements with positive probability.*

To prove this theorem we will need the following lemma.

Lemma 1 (Theorem 11.1.4 in [6])**.** *Consider a vector $\mathbf{X} \in \{0,1\}^n$ whose elements are i.i.d random variables sampled according to* Ber(p)*, $p < \frac{1}{2}$. The probability that more than $\frac{n}{2}$ are `1` is at least $\frac{2^{-nD(\frac{1}{2}||p)}}{(n+1)^2}$.*

Proof of Theorem 2. Suppose a querying scheme uses $n-a$ pairwise 'same cluster' queries. Consider a graph with n vertices corresponding to the querying scheme. The vertices are labeled $1, \ldots, n$, and the edge (i,j) exists if and only if (i,j) is a query. Since the number of edges in the graph is $n-a$, the graph has at least a components. Therefore average size of any component in the graph is $\frac{n}{a}$. This also implies that there exists at least $\frac{a}{2}$ components with size at most $\frac{2n}{a}$ each (using Markov inequality).

Consider the elements corresponding to vertices of one such component. Even if all the possible 'same cluster' queries were made within this group, we will still have two possible labelings that would be consistent with all possible query answers (for every assignment of labels, a new assignment can be created by flipping all the labels). Since this set of elements are not queried with any element

outside of it, this will give rise to 2^a different possibilities. This situation can only be mitigated if we can turn the clustering into labeling. Since there are two possible labelings within a group, we will make a mistake in labeling only when the number of elements with label 1 is larger than the number of elements with label 0 (the maximum likelihood decoding will fail).

Now, using Lemma 1, within a component of size at most $\frac{2n}{a}$, the probability that the number of elements with label 1 is less than the number of elements with label 0 is at most

$$1 - \frac{2^{-2nD(\frac{1}{2}||p)/a}}{(2n/a+1)^2}.$$

And the probability that this happens for all $\frac{a}{2}$ such components is at most

$$\Big(1 - \frac{2^{-2nD(\frac{1}{2}||p)/a}}{(2n/a+1)^2}\Big)^{a/2} \leq \exp\Big(-\frac{a2^{-2nD(\frac{1}{2}||p)/a}}{2(2n/a+1)^2}\Big) = o(1),$$

if we substitute $a = \frac{(2+\epsilon)nD(\frac{1}{2}||p)}{\log n}$ for any positive constant $\epsilon > 0$.

□

The main takeaway from this part is that, by exploiting the prior probabilities (or relative cluster sizes), it is possible to infer the labels with strictly less than n 'same cluster' queries. However, to make the reduction $O(n)$ we need to look at either a different type of querying, or involve more than two elements in a query.

3.1.2 XOR queries for larger Δ

Querying scheme: We use the following type of queries. For each query, labels of Δ elements are given to the oracle, and the oracle returns a simple XOR operation of the labels. Note, for $\Delta = 2$, our queries are just 'same cluster' queries. Let us define $\mathbf{Q}$ to be the binary query matrix of dimension $m \times n$ where each row has at most Δ ones, other entries being zero. Now for a label vector $\mathbf{X}$ we can represent the set of query outputs by $\mathbf{Y} = \mathbf{QX} \pmod 2$. In order to fulfill our objective, we will define a random ensemble of query matrices $\mathcal{Q}$ from which $\mathbf{Q}$ is sampled and subsequently, we will show that the average probability of error goes to zero asymptotically with n that implies the existence of a good matrix $\mathbf{Q}$ in the ensemble.

Random ensemble: The random ensemble $\mathcal{Q}$ will be defined in terms of a bipartite graph. This is done by constructing a biregular bipartite graph $G(V_1 \sqcup V_2, E)$, with $|V_1| = n, |V_2| = m$. Here V_1, called the left vertices, represents the labels; and V_2, the right vertices, represents the query. The degree of each left vertex is c and the degree of each right vertex is Δ. We have $|E| = m\Delta = nc$. Now, a permutation π is randomly sampled from the set of all permutation on $\{1, 2, \ldots, nc\}$, the symmetric group S_{nc}. If we fix an ordering of the edges emanating from vertices in the left and right, then ith edge will be joined with the $\pi(i)$th edge on the right.

Decoder: In this setting we will be concerned with the exact recovery of the labels. The decoder $\Psi : \{0,1\}^m \to \{0,1\}^n$ will look at the vector $\mathbf{QX}$ and return a vector $\hat{\mathbf{X}}$ such that the Hamming weight (number of nonzero entries) of the vector is given by $|\text{wt}(\hat{\mathbf{X}}) - np| \leq n^{2/3}$ and $\mathbf{Q}\hat{\mathbf{X}} = \mathbf{QX}$. Hence, the probability of error P_e can be defined as

$$P_e \equiv \sum_{\mathbf{X} \in \{0,1\}^n} \Pr(\mathbf{X}) \Pr_{\mathbf{Q} \sim \mathcal{Q}}(\Psi(\mathbf{QX}) \neq \mathbf{X}).$$

We have the following theorem

Theorem 3. *Consider an $m \times n$ query matrix $\mathbf{Q}$ sampled from the ensemble $\mathcal{Q}$ of matrices described above, and a label vector $\boldsymbol{X}$ with each entry being i.i.d.* Ber$(p), p < \frac{1}{2}$. *Let c, Δ be the left and the right degrees of the ensemble with $3 \leq c < \Delta$ and let $\beta = \frac{2}{\Delta}\Big(\frac{1}{2\Delta^2 p(1-p)e^{3c/2}}\Big)^{\frac{1}{c-2}}$. If the number of queries*

$$m > n \max_{\beta \leq x \leq 2p} \frac{pH\Big(\frac{x}{2p}\Big) + (1-p)H\Big(\frac{x}{2(1-p)}\Big)}{1 - \log\Big(1 + \Big(1 - 2x\Big)^{\Delta}\Big)},$$

then the average probability of error P_e goes to zero as

$$P_e \leq n^{2-c}(1-p+p^2)(\Delta c)^c(1+o(1)).$$

The proof of this theorem is delegated to Section 5. The same ensemble $\mathcal{Q}$ was used by [23] where the authors showed the existence of linear codes achieving zero probability of error in the Binary Symmetric Channel such that the parity check matrix of the code belonged to the ensemble $\mathcal{Q}$. Because of the duality of source-channel coding, their guarantees on the average probability of error for $\mathcal{Q}$ directly translate to our setting as well. However, our analysis is slightly different from [23] and it is tighter in most cases. We have delegated the detailed comparison of the two analyses to Section 5. The achievability result is depicted in Figure 1.

3.1.3 Lower bounds (converse)

Now we provide some necessary conditions on the number of queries, involving Δ elements at most, required for a full recovery of labels. First of all notice that, if a query involves at most Δ elements, then $\lceil \frac{n}{\Delta} \rceil$ queries are necessary for exact recovery. If a particular label is not present in any query, then the decoder has no other choice but to guess the label. This will lead to a constant probability of error $\min(p, 1-p)$. Therefore at least $\lceil \frac{n}{\Delta} \rceil$ queries are necessary so that every label is present in at least one query.

Adapting Gallager's result for low density parity-check matrix codes for our setting (using source-channel duality for linear codes) we can have the following lower bound on the number of queries.

Theorem 4 ([8])**.** *Assume a label vector $\boldsymbol{X}$ with each entry being i.i.d.* Ber$(p), p < \frac{1}{2}$*. If the number of XOR queries, each involving at most Δ labels, is less than $\frac{nH(p)}{H(\frac{1+(1-2p)^\Delta}{2})}$, then the probability of error in recovery of labels is bounded from below by a constant independent of the number of elements n.*

However Gallager's result is valid for only XOR queries. We can provide a lower bound that is close to Gallager's bound, and holds for any type of query function.

Theorem 5. *Assume a label vector $\boldsymbol{X}$ with each entry being i.i.d.* Ber$(p), p < \frac{1}{2}$*. The minimum number of queries, each involving at most Δ elements, necessary to recover all labels with high probability is at least by $nH(p) \cdot \max\{1, \max_\rho \frac{(1-\rho)}{H(\frac{(1-\rho)r(p)\Delta}{\rho})}\}$ where $r(p) \equiv 2p(1-p)$.*

Proof. Every query involves at most Δ elements. Therefore the average number of queries an element is part of is $\frac{\Delta m}{n}$. Therefore $1-\rho$ fraction of all the elements (say the set $S \subset \{1, \ldots, n\}$) are part of less than $\frac{\Delta m}{\rho n}$ queries. Now consider the set $\{1, \ldots, n\} \setminus S$. Consider all typical label vectors $\mathcal{C} \in \{0,1\}^n$ such that their projection on $\{1, \ldots, n\} \setminus S$ is a fixed typical sequence. We know that there are $2^{n(1-\rho)H(p)}$ such sequences. Let $\boldsymbol{X}_0$ be one of these sequences. Now, almost all sequences of $\mathcal{C}$ must have a distance of $n(1-\rho)r(p) + o(n)$ from $\boldsymbol{X}_0$. Let $\boldsymbol{Y}_0$ be the corresponding query outputs when $\boldsymbol{X}_0$ is the input. Now any query output for input belonging to $\mathcal{C}$ must reside in a Hamming ball of radius $\dfrac{(1-\rho)r(p)\Delta m}{\rho}$ from $\boldsymbol{Y}_0$. Therefore, comparing the volume of the balls, we must have $mH(\frac{(1-\rho)r(p)\Delta}{\rho}) \geqslant n(1-\rho)H(p)$. □

Finally, in Figure 1, we have compared the achievability scheme (Theorem 3) and the lower bounds presented above for $\Delta = 10$. For larger Δ the bounds are even closer.

3.2 Noiseless queries and approximate recovery

Again let us consider the case when $k = 2$, i.e., only two possible labels. Let $\boldsymbol{X} \in \{0,1\}^n$ be the label vector. Suppose we have a querying algorithm that, by using m queries, recovers a label vector $\hat{\boldsymbol{X}}$.

Definition. We call a querying algorithm to be $(1-\delta)$-good if for any label vector, at least $(1-\delta)n$ labels are correctly recovered. This means for any label-recovered label pair $\boldsymbol{X}, \hat{\boldsymbol{X}}$, the Hamming distance is at most δn. For an almost equivalent definition, we can define a distortion function

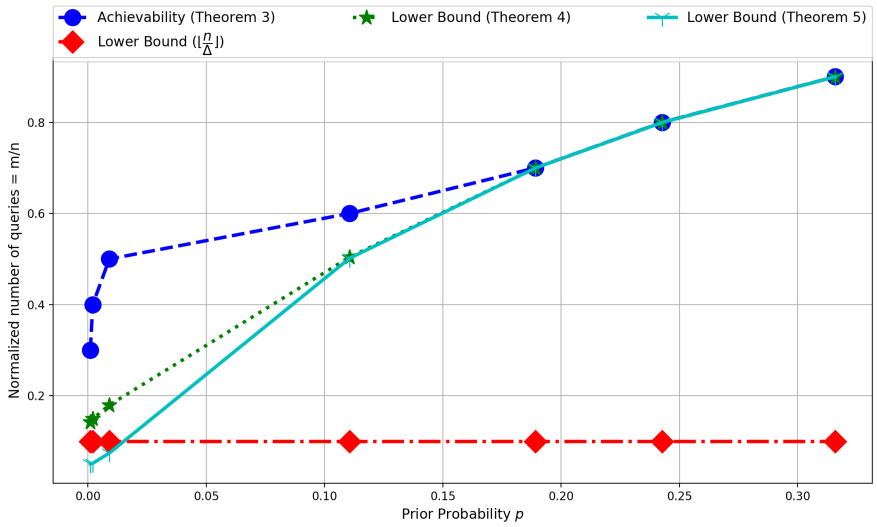


Figure 1: Normalized Query complexity $\frac{m}{n}$ of Theorem 3 (blue) as a function of prior probability p, for $\Delta = 10$.

$d(X, \hat{X}) = X + \hat{X} \mod 2$, for any two labels $X, \hat{X} \in \{0, 1\}$. We can see that $\mathbb{E}d(X, \hat{X}) = \Pr(X \neq \hat{X})$, which we want to be bounded by δ.

Using standard rate-distortion theory [6], it can be seen that, if the queries could involve an arbitrary number of elements then with m queries it is possible to have a $(1 - \tilde{\delta}(m/n))$-good scheme where $\tilde{\delta}(\gamma) \equiv h^{-1}(H(p) - \gamma)$. When each query is allowed to take only at most Δ inputs, we have the following lower bound for $(1 - \delta)$-good querying algorithms.

Theorem 6. *In any $(1 - \delta)$-good querying scheme with m queries where each query can take as input Δ elements, the following must be satisfied (below $h'(x) = \frac{dH(x)}{dx}$):*

$$\delta \geq \tilde{\delta}\left(\frac{m}{n}\right) + \frac{H(p) - H(\tilde{\delta}(\frac{m}{n}))}{h'(\tilde{\delta}(\frac{m}{n}))(1 + e^{\Delta h'(\tilde{\delta}(\frac{m}{n}))})}.$$

The proof of this theorem is somewhat involved, and we have provided it in Section 6.

One of the main observation that we make is that the 'same cluster' queries are highly inefficient for approximate recovery. This follows from a classical result of Ancheta and Massey [17] on the limitation of linear codes as rate-distortion codes. Recall that, the 'same cluster' queries are equivalent to XOR operation in the binary field, which is a linear operation on $GF(2)$. We rephrase a conjecture by Massey in our terminology.

Conjecture 1 ('same cluster' query lower bound)**.** *For any $(1 - \delta)$-good scheme with m 'same cluster' queries ($\Delta = 2$), the following must be satisfied: $\delta \geq p(1 - \frac{m}{nH(p)})$.*

This conjecture is known to be true at the point $p = 0.5$ (equal sized clusters). We have plotted these two lower bounds in Figure 2.

Now let us provide a querying scheme with $\Delta = 2$ that will provably be better than 'same cluster' schemes.

Querying scheme: AND queries: We define the AND query $Q : \{0, 1\}^2 \to \{0, 1\}$ as $Q(X, X') = X \bigwedge X'$, where $X, X' \in \{0, 1\}$, so that $Q(X, X') = 1$ only when both the elements have labels equal to 1. For each pairwise query the oracle will return this AND operation of the labels.

Theorem 7. *There exists a $(1 - \delta)$-good querying scheme with m pairwise AND queries such that*

$$\delta = pe^{-\frac{2m}{n}} + \sum_{d=1}^{n} \frac{e^{-\frac{2m}{n}}(\frac{2m}{n})^d}{d!} \sum_{k=1}^{d} \binom{n}{k} \frac{f(k, d)}{n^d} (1 - p)^k p$$

where $f(k,d) = \sum_{i=0}^{k}(-1)^i\binom{k}{i}(k-i)^d$.

Proof. Assume $p < 0.5$ without loss of generality. Consider a random bipartite graph where each 'left' node represent an element labeled according to the label vector $\boldsymbol{X} \in \{0,1\}^n$ and each 'right' node represents a query. All the query answers are collected in $\boldsymbol{Y} \in \{0,1\}^m$. The graph has right-degree exactly equal to 2. For each query the two inputs are selected uniformly at random without replacement.

Recovery algorithm: For each element we look at the queries that involves it and estimate its label as 1 if any of the query answers is 1 and predict 0 otherwise. If there are no queries that involves the element, we simply output 0 as the label.

Since the average left-degree is $\frac{2m}{n}$ and since all the edges from the right nodes are randomly and independently thrown, we can model the degree of each left-vertex by a Poisson distribution with the mean $\lambda = \frac{2m}{n}$. We define element j to be a two-hop-neighbor of i if there is at least one query which involved both the elements i and j . Under our decoding scheme we only have an error when the label of i, $X_i = 1$ and the labels of all its two-hop-neighbors are 0. Hence the probability of error for estimating X_i can be written as, $\Pr(X_i \neq \hat{X}_i) = \sum_d \Pr(\text{degree}(i) = d)\Pr(X_i \neq \hat{X}_i \mid \text{degree}(i) = d)$. Now let us estimate $\Pr(X_i \neq \hat{X}_i \mid \text{degree}(i) = d)$. We further condition the error on the event that there are k distinct two-hop-neighbors (lets call the number of distinct neighbors of i as $\text{Dist}(i)$) and hence we have that $\Pr(X_i \neq \hat{X}_i \mid \text{degree}(i) = d) = \sum_{k=1}^{d}\Pr(\text{Dist}(i) = k)\Pr(X_i \neq \hat{X}_i|\text{degree}(i) = d, \text{Dist}(i) = k) = \sum_{k=1}^{d}\binom{n}{k}\frac{f(k,d)}{n^d}p(1-p)^k$. Now using the Poisson assumption we get the statement of the theorem. □

The performance of this querying scheme is plotted against the number of queries for prior probabilities $p = 0.5$ in Figure 2.

Comparison with 'same cluster' queries: We see in Figure 2 that the AND query scheme beats the 'same cluster' query lower bound for a range of query-performance trade-off in approximate recovery for $p = \frac{1}{2}$. For smaller p, this range of values of δ increases further. If we randomly choose 'same cluster' queries and then resort to maximum likelihood decoding (note that, for AND queries, we present a simple decoding) then $O(n \log n)$ queries are still required even if we allow for δ proportion of incorrect labels (follows from [11]). The best performance for 'same cluster' query in approximate recovery that we know of for small δ is given by: $m = n(1-\delta)$ (neglect $n\delta$ elements and just query the $n(1-\delta)$ remaining elements with the first element). However, such a scheme can be achieved by AND queries as well in a similar manner. Therefore, there is no point in the query vs δ plot that we know of where 'same cluster' query achievability outperforms AND query achievability.

3.3 Noisy queries and approximate recovery

This section contains our main algorithmic contribution. In contrast to the previous sections here we consider the general case of $k \geq 2$ clusters. Recall that the label vector $\boldsymbol{X} \in \{0,1,\ldots,k-1\}^n$, and the prior probability of each label is given by the probabilities $\boldsymbol{p} = (p_0,\ldots,p_{k-1})$. Instead of binary output queries, in this part we consider an oracle that can provide one of k different answers. We consider a model of noise in the query answer where the oracle provides correct answer with probability $1-q$, and any one of the remaining incorrect answers with probability $\frac{q}{k-1}$. Note that we do not allow the same query to be asked to the oracle multiple time (see Sec. 2 for justification). We also define a $(1-\delta)$-good approximation scheme exactly as before.

Querying Scheme: We only perform pairwise queries. For a pair of labels X and X' we define a query $Y = Q(X,X') \in \{0,1,\ldots,k-1\}$. For our algorithm we define the Q as

$$Q(X,X') = \left\{ \begin{array}{ll} i & \textit{if} \quad X = X' = i \\ 0 & \text{otherwise.} \end{array} \right\}$$

We can observe that for $k = 2$, this query is exactly same as the binary AND query that we defined in the previous section. In our querying scheme, we make a total of $\frac{nd}{2}$ queries, for an integer $d > 1$. We design a d-regular graph $G(V,E)$ where $V = \{1,\ldots,n\}$ is the set of elements that we need to label. We query all the pairs of elements $(u,v) \in E$.

Under this querying scheme, we propose to use Algorithm 1 for reconstructions of labels.

Theorem 8. *The querying scheme with $m = O(n \log \frac{k}{\delta})$ queries and Algorithm 1 is $(1-\delta)$-good for approximate recovery of labels from noisy queries.*

Algorithm 1 Noisy query approximate recovery with $\frac{nd}{2}$ queries

Require: PRIOR $\boldsymbol{p} \equiv (p_0, \ldots, p_{k-1})$
Require: Query Answers $Y_{u,v} : (u,v) \in E$
  for $i \in [1, 2, \ldots, k-1]$ **do**
    $C_i = \frac{dq}{k-1} + \frac{dp_i}{2}\big(1 - \frac{qk}{k-1}\big)$
  end for
  for $u \in V$ **do**
    for $i \in [1, 2, \ldots, k-1]$ **do**
      $N_{u,i} = \sum_{v=1}^{d} 1\{Y_{u,v} = i\}$
      if $N_{u,i} \geq \lceil C_i \rceil$ **then**
        $X_u \leftarrow i$
        Assigned $\leftarrow$ True
        break
      end if
    end for
    if $\neg$ Assigned **then**
      $X_u \leftarrow 0$
    end if
  end for

We can come up with more exact relation between number of queries $m = \frac{nd}{2}$, δ, p, q and k. This is deferred to Section 7.

Proof of Theorem 8. The total number of queries is $m = \frac{nd}{2}$. Now for a particular element $u \in V$, we look at the values of d noisy oracle answers $\{Y_{u,v}\}_{v=1}^{d}$. We have, $\mathbb{E}(N_{u,i}) = \frac{dq}{k-1} + dp_i\big(1 - \frac{qk}{k-1}\big)$ when the true label of u is $i \neq 0$. When the true label is something else, $\mathbb{E}(N_{u,i}) = \frac{dq}{k-1}$. There is an obvious gap between these expectations. Clearly when the true label is i, the probability of error in assignment of the label of u is given by, $P_i \leq \sum_{j:j\neq i, j\neq 0} \Pr(N_{u,j} > C_j) + \Pr(N_{u,i} < C_i) \leq cke^{-2d\epsilon^2}$ for some constants c and ϵ depending on the gap, from Chernoff bound. And when the true label is 0, the probability of error is $P_0 \leq \sum_{j:j\neq 0} P(N_{u,j} > C_j) \leq c'ke^{-2d\epsilon'^2}$, for some constants c', ϵ'. Let $\delta = \sum_i p_i P_i$, we can easily observe that d scales as $O(\log \frac{k}{\delta})$. Hence the total number of queries is $\frac{nd}{2} = O(n \log \frac{k}{\delta})$.

The only thing that remains to be proved is that the number of incorrect labels is δn with high probability. Let Z_u be the event that element u has been incorrectly labeled. Then $\mathbb{E}Z_u = \delta$. The total number of incorrectly labeled elements is $Z = \sum_u Z_u$. We have $\mathbb{E}Z = n\delta$. Now define $Z_u \sim Z_v$ if Z_u and Z_v are dependent. Now $\Delta^* \equiv \sum_{Z_u \sim Z_v} \Pr(Z_u | Z_v) \leq d^2 + d$ because the maximum number of nodes dependent with Z_u are its 1-hop and 2-hop neighbors. Now using Corollary 4.3.5 in [2], it is evident that $Z = \mathbb{E}Z = n\delta$ almost always. □

The theoretical performance guarantee of Algorithm 1 (a detailed version of Theorem 8 is in the supplementary material) for $k = 2$ is shown in Figures 3 and 4. We can observe from Figure 3 that for a particular q, incorrect labeling decreases as p becomes higher. We can also observe from Figure 4 that if $q = 0.5$ then the incorrect labeling is 50% because the complete information from the oracle is lost. For other values of q, we can see that the incorrect labeling decreases with increasing d.

We point out that 'same cluster' queries are not a good choice here, because of the symmetric nature of XOR due to which there is no gap between the expected numbers (contrary to the proof of Theorem 8) which we had exploited in the algorithm to a large extent.

Lastly, we show that Algorithm 1 can work without knowing the prior distribution and only with the knowledge of relative sizes of the clusters. The ground truth clusters can be adversarial as long as they maintain the relative sizes.

Theorem 9. *Suppose we have n_i, the number of elements with label i, $i = 0, 1, \ldots, k-1$, as input instead of the priors. By taking a random permutation over the nodes while constructing the d-regular graph, Algorithm 1 will be $(1-\delta)$-good approximation with $m = O(n \log \frac{k}{\delta})$ queries as $n \to \infty$ when we set $p_i = \frac{n_i}{n}$.*

Proof. Let us consider $k = 2$. The proof can be extended for general k. We generate the d-regular graph on a n nodes is the following way:

1. If $d = 2\ell$ is even, put all the vertices around a circle, and join each to its ℓ nearest neighbors on either side.

2. If $d = 2\ell + 1$ is odd, and n is even, put the vertices on a circle, join each to its ℓ nearest neighbors on each side, and also to the vertex directly opposite.

Now suppose we are given n_0 and n_1. Let us consider all random permutations of these sets of points on a circle. Fixing a node u of label 1(say), it becomes a random permutation on a line by making u a reference point. The probability that there are exactly k neighbors of u having label 1 among the d neighbors is

$$\binom{d}{k}\frac{\binom{n-d-1}{n_1-k-1}}{\binom{n-1}{n_1-1}} \approx \binom{d}{k}\frac{\Pi_{i=0}^{k-1}(n_1-i)\Pi_{i=0}^{d-k-1}(n_0-i)}{\Pi_{i=0}^{d-1}(n-i)} \leq \binom{d}{k}(\frac{n_1}{n})^k(\frac{n_0}{n-k})^{d-k}.$$

All the rest of the conditional probabilities used in the analysis in Appendix 7 stays the same. Now, k is just a constant $\leq d$ and hence $n - k \approx n$ and hence asymptotically this distribution is equivalent to the binomial distribution. Hence we can simply set $C = dq + \frac{dn_1}{2n}(1-2q)$ and then use the Algorithm 1. Our final probability of incorrect labeling is going to be $\delta = \frac{n_1}{n}P(X \neq \hat{X}|X = 1) + \frac{n_0}{n}P(X \neq \hat{X}|X = 0)$. Thus for large n its behavior is exactly the same as with using priors. □

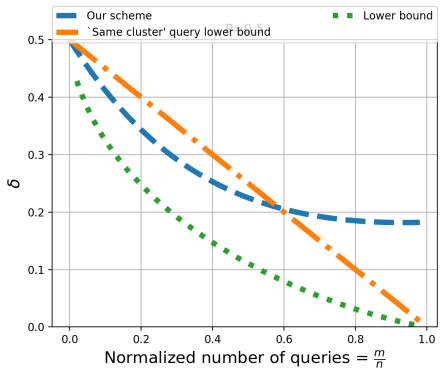


Figure 2: Performance of $(1-\delta)$-good schemes with noiseless queries; $p = 0.5$

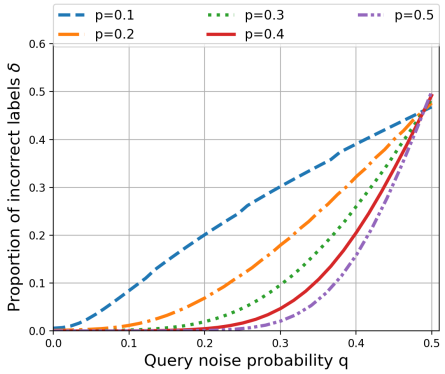


Figure 3: Recovery error for a fixed $p, d = 100$ and varying q

4 Experiments

Though our main contribution is theoretical we have verified our work by using our algorithm on a real dataset created by local crowdsourcing. We first picked a list of 100 'action' movies and 100 'romantic' movies from IMDB (`http://www.imdb.com/list/ls076503982/` and `http://www.imdb.com/list/ls058479560/`). We then created the queries as given in the

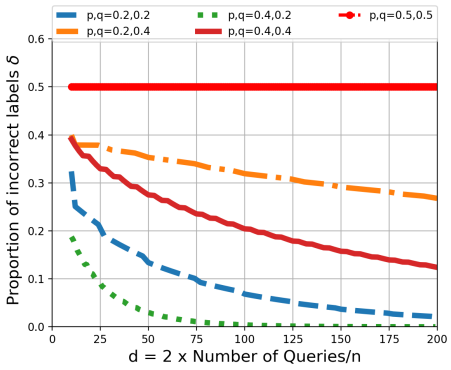


Figure 4: Recovery error for a fixed p, q and varying d

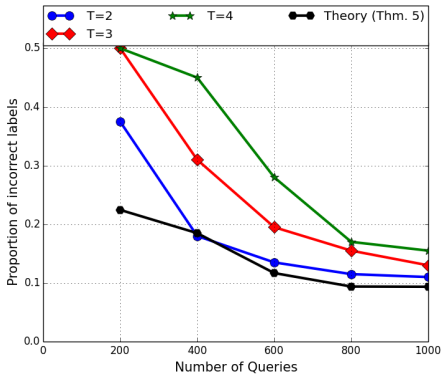


Figure 5: Algorithm 1 on real crowdsourced dataset

querying scheme of Sec. 3.3 by creating a d-regular graph (where d is even). To create the graph we put all the movies on a circle and took a random permutation on them in a circle. Then for each node we connected $\frac{d}{2}$ edges on either side to its closest neighbors in the permuted circular list. This random permutation will allow us to use the relative sizes of the clusters as priors as explained in Sec. 3.3. Using $d = 10$, we have $\frac{nd}{2} = 1000$ queries with each query being the following question: *Are both the movies 'action' movies?*. Now we divided these 1000 queries into 10 surveys (using SurveyMonkey platform) with each survey carrying 100 queries for the user to answer. We used 10 volunteers to fill up the survey. We instructed them not to check any resources and answer the questions spontaneously and also gave them a time limit of a maximum of 10 minutes. The average finish time of the surveys were 6 minutes. The answers represented the noisy query model since some of the answers were wrong. In total, we have found 105 erroneous answers in those 1000 queries. For each movie we evaluate the d query answer it is part of, and use different thresholds T for prediction. That is, if there are more than T 'yes' answers among those d answers we classified the movie as 'action' movie and a 'romantic' movie otherwise.The theoretical threshold for predicting an 'action' movie is $T = 2$ for oracle error probability $q = 0.105, p = 0.5$ and $d = 10$. But we compared other thresholds as well. We now used Algorithm 1 to predict the true label vector from a subset of queries by taking $\tilde{d}$ edges for each node where $\tilde{d} < d$ and $\tilde{d}$ is even i.e $\tilde{d} \in \{2, 4, 6, 8, 10\}$. Obviously, for $\tilde{d} = 2$, the thresholds $T = 3, 4$ is meaningless as we always estimate the movie as 'romantic' and hence the distortion starts from 0.5 in that case. We plotted the error for each case against the number of queries ($\frac{n\tilde{d}}{2}$) and also plotted the theoretical distortion obtained from our results for $k = 2$ labels and $p = 0.5, q = 0.105$. We compare these results along with the theoretical distortion that we should have for $q = 0.105$. All these results have been compiled in Figure 5 and we can observe that the distortion is decreasing with the number of queries and the gap between the theoretical result and the experimental results is small for $T = 2$. These results validate our theoretical results and our algorithm to a large extent.

We have also asked 'same cluster' queries with the same set of 1000 pairs to the participants to find that the number of erroneous responses to be 234 (whereas with AND queries it was 105). This substantiates the claim that AND queries are easier to answer for workers. Since this number of errors is too high, we have compared the performance of 'same cluster' queries with AND queries and our algorithm in a synthetically generated dataset with two hundred elements (Figure 6). For recovery with 'same cluster' queries, we have used the popular spectral clustering algorithm with normalized cuts [25]. The detailed results obtained can be found in Figure 7 below.

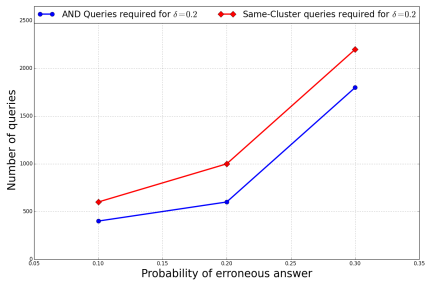


Figure 6: Comparison of 'same cluster' query with AND queries when both achieve 80% accuracy

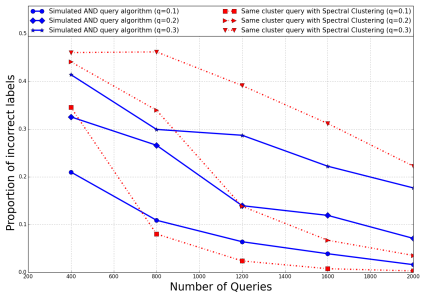


Figure 7: Comparison of performance of 'same cluster' query with AND queries on a randomly generated dataset for varying probability of erroneous answers and varying number of queries. The AND querying methods performs well with higher probability of erroneous answers.

5 Proof of Theorem 3 and comparison with [23]

We have assumed that the label vector $\boldsymbol{X} \in \{0, 1\}^n$ consists of i.i.d. Bernoulli random variables with $\Pr(1) = p < \frac{1}{2}$. We will define the typical set $\text{Typ}(p) \equiv \{\mathbf{Z} \in \{0, 1\}^n \mid np - n^{2/3} \leq \text{wt}(\mathbf{Z}) \leq np + n^{2/3}\}$, where $\text{wt}(\cdot)$ denote the Hamming weight of the argument. We know that (See [6]) $\Pr(\mathbf{X} \in \text{Typ}(p)) \geq 1 - \epsilon_n$, with $\epsilon_n \to 0$ as $n \to \infty$. For a fixed vector $\mathbf{X}$ in the typical

set, let us denote $A_{t,\mathbf{X}}$ as the number of vectors in the typical set that are at a distance t from $\mathbf{X}$. Let us denote the weight of the vector $\mathbf{X}$ by $np+s$ where $|s| \leq n^{2/3}$. In that case, it is easy to see that $A_{t,\mathbf{X}} = \sum_{j:|j|\leq 2n^{2/3}} \binom{np+s}{\frac{t}{2}-j}\binom{n-np-s}{\frac{t}{2}+j}$[4]. Let A_t be defined as $\max_{\mathbf{X}\in \text{Typ}(p)} A_{t,\mathbf{X}}$ which is independent of the vector $\mathbf{X}$ and suppose that the weight of the vector $\mathbf{X}$ which maximizes $A_{t,\mathbf{X}}$ is $np+s^*$ for some $s^* : |s^*| \leq n^{2/3}$. For two binary vectors $\mathbf{X}, \mathbf{Y} \in \{0,1\}^n$ we will denote $d_H(\mathbf{X},\mathbf{Y})$ to be the Hamming distance between the two vectors. Now, we can rewrite the probability of error P_e as below:

$$
\begin{aligned}
P_e &= \sum_{\mathbf{X}\in \text{Typ}(p)} \Pr(\mathbf{X}) \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\Psi(\mathbf{QX}) \neq \boldsymbol{X}) + \sum_{\boldsymbol{X}\notin \text{Typ}(p)} \Pr(\boldsymbol{X}) \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\Psi(\mathbf{Q}\boldsymbol{X}) \neq \boldsymbol{X}) \\
&\leq \sum_{\boldsymbol{X}\in \text{Typ}(p)} \Pr(\boldsymbol{X}) \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\Psi(\mathbf{Q}\boldsymbol{X}) \neq \boldsymbol{X}) + \sum_{\boldsymbol{X}\notin \text{Typ}(p)} \Pr(\boldsymbol{X}) \\
&= \sum_{\boldsymbol{X}\in \text{Typ}(p)} \Pr(\boldsymbol{X}) \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\exists \boldsymbol{X}' \in \text{Typ}(p) \text{ such that } \mathbf{Q}\boldsymbol{X} = \mathbf{Q}\mathbf{X}') + \epsilon_n \\
&\leq \sum_{\boldsymbol{X}\in \text{Typ}(p)} \Pr(\boldsymbol{X}) \sum_{t=1}^{2np+2n^{2/3}} \sum_{\mathbf{X}'\in\text{Typ}(p)|d_H(\boldsymbol{X},\boldsymbol{X}')=t} \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{Q}\boldsymbol{X} = \mathbf{Q}\boldsymbol{X}') + \epsilon_n \\
&= \sum_{\boldsymbol{X}\in \text{Typ}(p)} \Pr(\boldsymbol{X}) \sum_{t=1}^{2np+2n^{2/3}} A_{t,\boldsymbol{X}} \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) + \epsilon_n \\
&\leq \sum_{t=1}^{2np+2n^{2/3}} A_t \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) + \epsilon_n \\
&\leq \sum_{t \text{ odd}, t=1}^{2np+2n^{2/3}} A_t \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) + \sum_{t \text{ even}, t=2}^{2np+2n^{2/3}} A_t \Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) + \epsilon_n \\
&\leq P_e^{\text{odd}} + P_e^{\text{even}} + \epsilon_n,
\end{aligned}
$$

where the first two terms in the sum are termed P_e^{odd} and P_e^{even} respectively. Now, we will use the following Lemma from [23]

Lemma 2. *If tc is odd, then*

$$\Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) = 0$$

otherwise, we will have the following two upper bounds:

$$
\begin{aligned}
\Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) &\leq \binom{m}{\frac{tc}{2}} \Big(\frac{tc}{2m}\Big)^{tc} \quad \text{for } t \leq \frac{2m}{c}, \\
\Pr_{\mathbf{Q}\sim\mathcal{Q}}(\mathbf{QZ} = 0 \mid \text{wt}(\mathbf{Z}) = t) &\leq (m\Delta+1)2^{-m}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^m.
\end{aligned}
$$

Therefore, after substituting the values of A_t, we can write:

$$
\begin{aligned}
P_e^{\text{even}} \leq &\sum_{t \text{ even}, t=2}^{\beta n} \binom{m}{\frac{tc}{2}} \Big(\frac{tc}{2m}\Big)^{tc} \sum_j \binom{np+s^*}{\frac{t}{2}-j}\binom{n-np-s^*}{\frac{t}{2}+j} + \\
&\sum_{t \text{ even}, t=\beta n}^{2np+2n^{2/3}} \frac{m\Delta+1}{2^m}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^m \sum_j \binom{np+s^*}{\frac{t}{2}+j}\binom{n-np-s^*}{\frac{t}{2}-j}
\end{aligned}
$$

[4]This is for the case when t is even. For odd t, we will have $A_{t,\boldsymbol{X}} = \sum_j \binom{np+s}{\frac{t-1}{2}-j}\binom{n-np-s}{\frac{t+1}{2}+j} + \binom{np+s}{\frac{t+1}{2}-j}\binom{n-np-s}{\frac{t-1}{2}+j}$

for some $0 \leq \beta \leq \frac{2}{\Delta}$ to be determined later. Let us define the function $f : \mathbb{Z} \to \mathbb{R}$ which takes a positive even number as input:

$$f(x) = \sum_j \binom{np+s^*}{\frac{x}{2}-j}\binom{n-np-s^*}{\frac{x}{2}+j}\binom{m}{\frac{cx}{2}}\Big(\frac{cx}{2m}\Big)^{cx}.$$

Notice that

$$\begin{aligned}
\frac{f(x+2)}{f(x)} &= \frac{\sum_j \binom{np+s^*}{\frac{x}{2}-j+1}\binom{n-np-s^*}{\frac{x}{2}+j+1}\binom{m}{\frac{cx}{2}+c}\Big(\frac{cx+2c}{2m}\Big)^{cx+2c}}{\sum_j \binom{np+s^*}{\frac{x}{2}-j}\binom{n-np-s^*}{\frac{x}{2}+j}\binom{m}{\frac{cx}{2}}\Big(\frac{cx}{2m}\Big)^{cx}} \\
&\leq \frac{\binom{m}{\frac{cx}{2}+c}\Big(\frac{cx+2c}{2m}\Big)^{cx+2c}}{\binom{m}{\frac{cx}{2}}\Big(\frac{cx}{2m}\Big)^{cx}} \cdot \max_j \frac{\binom{np+s^*}{\frac{x}{2}-j+1}\binom{n-np-s^*}{\frac{x}{2}+j+1}}{\binom{np+s^*}{\frac{x}{2}-j}\binom{n-np-s^*}{\frac{x}{2}+j}} \\
&= \max_j \Big(\frac{np+o(n)}{x/2-j+1} \cdot \frac{n-np-o(n)}{x/2+j+1}\Big) \cdot \frac{\prod_{i=1}^{c}(m-cx/2-i)}{\prod_{i=1}^{c}(cx/2+i)} \cdot \frac{\Big(\frac{cx+2c}{2m}\Big)^{cx+2c}}{\Big(\frac{cx}{2m}\Big)^{cx+2c-2c}} \\
&< \frac{np(n-np)+o(n^2)}{(\frac{x}{2}+1)^2} \cdot \Big(\frac{2m}{cx}-1\Big)^c \cdot \Big(1+\frac{2}{x}\Big)^{cx+2c} \cdot \Big(\frac{cx}{2m}\Big)^{2c} \\
&< \frac{np(n-np)+o(n^2)}{(\frac{x}{2}+1)^2} \cdot \Big(\frac{2m}{cx}-1\Big)^c \cdot \Big(1+\frac{2}{x}\Big)^{3cx} \cdot \Big(\frac{cx}{2m}\Big)^{2c} \\
&< \frac{(np(n-np)+o(n^2))e^{3c/2}}{(\frac{x}{2}+1)^2} \cdot \Big(\frac{cx}{2m}\Big)^c \\
&< \frac{(np(n-np)+o(n^2))e^{3c/2}}{x^2/4} \cdot \Big(\frac{cx}{2m}\Big)^c \\
&< \frac{c^2(np(n-np)+o(n^2)e^{3c/2}}{m^2} \cdot \Big(\frac{cx}{2m}\Big)^{c-2} \\
&< \frac{1}{2},
\end{aligned}$$

when $x < \frac{2n}{\Delta}\Big(\frac{1}{2\Delta^2 p(1-p)e^{3c/2}}\Big)^{\frac{1}{c-2}}$ (and using the fact $nc = m\Delta$). Hence, for $\beta = \frac{2}{\Delta}\Big(\frac{1}{2\Delta^2 p(1-p)e^{3c/2}}\Big)^{\frac{1}{c-2}}$, the condition $\frac{f(x+2)}{f(x)} < \frac{1}{2}$ is satisfied for all $x < \beta n$. In that case we can rewrite P_e^{even} as

$$P_e^{\text{even}} \leq 2f(2) + \sum_{t \text{ even},\, t=\beta n}^{2np+2n^{2/3}} \frac{m\Delta+1}{2^m}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^m \sum_j \binom{np+s^*}{\frac{t}{2}+j}\binom{n-np-s^*}{\frac{t}{2}-j} \quad (1)$$

where $f(2) \leq n^2\Big(p(1-p)+p^2+(1-p)^2+o(1)\Big)\binom{m}{c}\Big(\frac{c}{m}\Big)^{2c} \leq n^{2-c}\Big(p(1-p)+p^2+(1-p)^2+o(1)\Big)(\Delta c)^c$.

Let us denote by σ the second term of the right hand side in Eq. (1) which we can upper bound as follows:

$$\begin{aligned}
\sigma &= \sum_{t=\beta n: t \text{ is even}}^{2np+2n^{2/3}} \sum_j \binom{np+s^*}{\frac{t}{2}+j}\binom{n-np-s^*}{\frac{t}{2}-j} \frac{m\Delta+1}{2^m}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^m \\
&\leq \sum_{t=\beta n: t \text{ is even}}^{2np+2n^{2/3}} 2^{npH(\frac{t}{2np})+(n-np)H(\frac{t}{2n(1-p)})} \frac{nc+1}{2^{nc/\Delta}}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^{nc/\Delta}
\end{aligned}$$

where we have used the fact that $\sum_{j|\frac{j}{a}=o(1)} \binom{a}{\gamma a+j} \leq 2^{aH(\gamma)}$. Hence, on taking logarithm on both sides, we will have

$$\frac{\log \sigma}{n} \leq \max_{\beta \leq x \leq 2p} pH\Big(\frac{x}{2p}\Big) + (1-p)H\Big(\frac{x}{2(1-p)}\Big) + \frac{c}{\Delta}\log\Big(1+\Big(1-2x\Big)^{\Delta}\Big) - \frac{c}{\Delta} + \frac{o(n)}{n}$$

Therefore, if $\max_{\beta \leq x \leq 2p} pH\Big(\frac{x}{2p}\Big) + (1-p)H\Big(\frac{x}{2(1-p)}\Big) + \frac{c}{\Delta}\log\Big(1+\Big(1-2x\Big)^{\Delta}\Big) < \frac{c}{\Delta}$, then σ decreases exponentially with n. In the same way as above, we can also bound P_e^{odd} as follows:

$$P_e^{\text{odd}} \leq 2f(1) + \sum_{t=\beta n: t\text{odd}}^{2np+2n^{2/3}} \frac{m\Delta+1}{2^m}\Big(1+\Big(1-\frac{2t}{n}\Big)^{\Delta}\Big)^m \sum_j \binom{np+s^*}{\frac{t}{2}+j}\binom{n-np-s^*}{\frac{t}{2}-j} \quad (2)$$

Now the first term is small compared to $f(2)$ (i.e. $f(1) = o(f(2))$) and the second term again goes to zero asymptotically with n when the condition $\max_{\beta \leq x \leq 2p} pH\Big(\frac{x}{2p}\Big) + (1-p)H\Big(\frac{x}{2(1-p)}\Big) + \frac{c}{\Delta}\log\Big(1+\Big(1-2x\Big)^{\Delta}\Big) < \frac{c}{\Delta}$ is satisfied. Substituting $\frac{m}{n} = \frac{c}{\Delta}$ in the above condition, we get back the statement of Theorem 3.

Here we state the results of [23] in our notations:

Theorem 10 (Theorem 3, [23])**.** *Consider the random ensemble $\mathcal{Q}$ of Query matrices $\mathbf{Q}_{m\times n}$ described above and binary source* Ber(p) *from which n labels are sampled independently. Let c, Δ be the left degree and the right degree of the ensemble such that $3 \leq c < \Delta$ and let $\beta = \frac{2}{\Delta}\exp(-12 - \frac{6\ln\Delta}{c})$. If there exists $0 \leq \gamma \leq \frac{1}{2}$ such that*

$$\max_{\beta \leq x \leq \gamma} \frac{x\log\sqrt{2p(1-p)} + H(x)}{1-\log\Big(1+\Big(1-2x\Big)^{\Delta}\Big)} < \frac{m}{n}$$

and

$$\frac{m}{n} > \frac{H(p)}{1-\log\Big(1+\Big(1-2\gamma\Big)^{\Delta}\Big)}$$

then the average probability of error P_e goes to zero asymptotically.

It is difficult to compare analytically the bound of Thm. 10 with our result (Thm. 3). Note that, the number of queries is an increasing function of the prior probability (p). We have plotted the number of queries against the prior probabilities obtained from these two theorems for $\Delta = 7, 10$. It can be seen that in some points, our analysis (blue dots) is tighter than the expression in [23] (red dots), while being same/similar in others.

6 Proof of Theorem 6

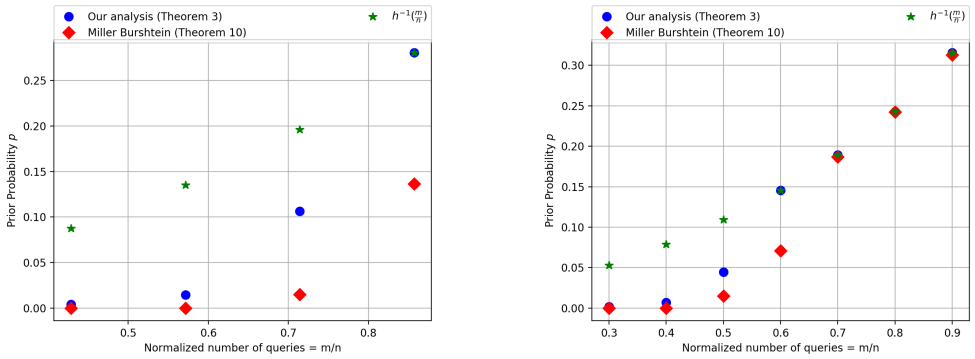
We will need the following definitions for this proof inspired by [5].

Definition. If the number of queries is m and the number of input labels is n then we define rate as the relative number of queries or $R = \frac{m}{n}$

Definition. The Rate-Distortion function $R(\delta)$ is the infimum of the feasible rates such that the scheme is $(1-\delta)$-good.

Definition. The Distortion-Rate function $\delta(R)$ is the infimum of all δ, for $(1-\delta)$-good schemes, when the rate is R.

Definition. The set of reconstructed label vectors are called *codewords*. Since the rate is R, our problem is to define the querying scheme $Q : \{0,1\}^n \to \{0,1\}^{nR}$ and a recovery $\{0,1\}^{nR} \to \{0,1\}^n$. We have a bijective mapping from query answers $\boldsymbol{Y}$ to $\hat{\boldsymbol{X}}$. Hence the total number of possible codewords is 2^{nR}.



(a) Comparison of Theorem 3 and 10 for $\Delta = 7$.

(b) Comparison of Theorem 3 and 10 for $\Delta = 10$.

Figure 8: Comparison of our analysis and the analysis in [23] for the random ensemble $\mathcal{Q}$

Proof of Theorem 6. We are interested in finding a lower bound on the distortion that we will achieve if we use a rate $R(\tilde{\delta})$ for the model where $R(\tilde{\delta})$ is the minimum rate for distortion $\tilde{\delta}$ achieved optimally in the unconstrained case. Now suppose that the distortion achieved in the model at rate $R(\tilde{\delta})$ is $\delta = \tilde{\delta} + \epsilon$ and hence we want a lower bound on ϵ . Since our input labels are typical sequences and it is mapped to a unique codeword, the reconstructed sequence must be having a per symbol distortion of less than $\tilde{\delta} + \epsilon$. We will be counting the number of label vector-codeword pairs (S, T) where $S \in \{0,1\}^n$ is a label vector and $T \in \{0,1\}^n$ is the corresponding codeword for S. Let us allow a small extra distortion of γ. Now, we have 2 ways to count the number of possible pairs. Firstly, from the perspective of the codewords , the number of possible pairs will be $2^{nR}\text{Vol}(\tilde{\delta} + \epsilon + \gamma)$ where $\text{Vol}(\tilde{\delta} + \epsilon + \gamma)$ is the number of label sequences present in the ball of radius $n(\tilde{\delta} + \epsilon + \gamma)$ from a particular codeword. Since there might be repetitions hence we are overcounting and hence this value is definitely an upper bound on the number of pairs. Again we can try to see from the perspective of the label sequences. Now let us say that we have a label sequence S and a corresponding compressed sequence C and the codeword T when no extra distortion is allowed. We will be trying to find out the number of other different codewords S could have mapped to when this extra distortion is allowed. Let us take ball of $n\gamma$ around S and take another label sequence in that ball and call it $\hat{S}$. Let the codeword it was initially mapped to be $\hat{T}$ Now,

$$|\hat{T} - S| \leq |\hat{T} - \hat{S}| + |\hat{S} - S| \leq n(\tilde{\delta} + \epsilon + \gamma)$$

Hence $\hat{T}$ is a possible candidate codeword for S if we allow this extra bit of more distortion γ. Hence we want a lower bound on the number of possible different codewords that can be candidate codewords for S when this extra distortion is allowed. Now we know that there is a bijective mapping from the compressed sequences to the codewords. Let x denote the fraction of bits in C that we can perturb. Then if, $nRx\Delta < n\gamma$ or $x < \frac{\gamma}{R\Delta}$, then there exists label vectors mapped to those compressed sequences which will be within a ball of $n\gamma$ from S and all the codewords corresponding to those perturbed compressed sequences must be different because of the bijective mapping. Since the total number of label sequences is $2^{nH(p)}$, the total number of pairs that can be calculated in such a way will be $2^{nH(p)+nRH(\frac{\gamma}{R\Delta})}$ which is a lower bound on the actual number of pairs. Since $\frac{1}{n}\log(Vol(\tilde{\delta} + \epsilon + \gamma)) = H(p) - R(\tilde{\delta} + \epsilon + \gamma)$ we have

$$R(\tilde{\delta}) - R(\tilde{\delta} + \epsilon + \gamma) \geqslant RH(\frac{\gamma}{R\Delta})$$

Using the fact that $R(\delta) = H(p) - H(\delta)$, we have

$$H(\tilde{\delta} + \epsilon + \gamma) - H(\tilde{\delta}) \geq RH(\frac{\gamma}{R\Delta}).$$

Now since entropy is a concave function of the distribution we must have $H(\tilde{\delta} + \epsilon + \gamma) \geq H(\tilde{\delta}) + (\epsilon + \gamma)h'(\tilde{\delta})$ where $h'(x) = \log\frac{(1-x)}{x}$ is the derivative of the binary entropy function. Plugging it into the formula, we have

$$\epsilon h'(\tilde{\delta}) \geq RH(\frac{\gamma}{R\Delta}) - \gamma h'(\tilde{\delta})$$

Now we want the value of γ in order to get the tightest lower bound. Hence, differentiating w.r.t γ and setting it to 0 in order to maximize it, we will have $\gamma = \frac{R\Delta}{1+e^{\Delta h'(\tilde{\delta})}}$ Using this value of γ, we plug in the original equation and we get

$$\epsilon \geq -\frac{R\Delta}{1+e^{\Delta h'(\tilde{\delta})}} + \frac{1}{h'(\tilde{\delta})} RH(\frac{1}{1+e^{\Delta h'(\tilde{\delta})}})$$

Expanding the entropy function we have

$$\epsilon \geq -\frac{R\Delta}{1+e^{\Delta h'(\tilde{\delta})}} + \frac{R}{h'(\tilde{\delta})(1+e^{\Delta h'(\tilde{\delta})})} \log(1+e^{\Delta h'(\tilde{\delta})}) + \frac{Re^{\Delta h'(\tilde{\delta})}}{h'(\tilde{\delta})(1+e^{\Delta h'(\tilde{\delta})})} \log(1+\frac{1}{e^{\Delta h'(\tilde{\delta})}})$$

Now if use the facts that $\log(1+e^{\Delta h'(\tilde{\delta})}) \geq \Delta h'(\tilde{\delta})$, $\log(1+\frac{1}{e^{\Delta h'(\tilde{\delta})}}) \geq \frac{1}{e^{\Delta h'(\tilde{\delta})}}$ and $R(\tilde{\delta}) = H(p) - H(\tilde{\delta})$, we get that

$$\epsilon \geq \frac{H(p) - H(\tilde{\delta})}{h'(\tilde{\delta})(1+e^{\Delta h'(\tilde{\delta})})}.$$

□

7 Exact Relation between δ, p, q, k and d in Theorem 8

In the scheme that when the true label of u is $i \neq 0$, $\mathbb{E}[N_{u,i}] = \frac{dq}{k-1} + dp_i\big(1 - \frac{qk}{k-1}\big)$, and when it is anything else, $\mathbb{E}[N_{u,i}] = \frac{dq}{k-1}$. This justifies the threshold that we have described. Now let us calculate the probability of error under this scheme. It is evident that the error is symmetric $\forall i \neq 0$. Now there are two kinds of errors that are possible when the true label is i. The first type of error occurs when $N_{u,j} > C_j$ for some $j \neq i, 0$ and the second type of error occurs when $N_{u,i} < C_i$. When the true label is $i \neq 0$, the first type of decoding error can be written as follows

$$\Pr(N_{u,k} > C_k | k \neq 0, k \neq i) = \sum_{j=\lceil C_k \rceil}^{d} \binom{d}{j} (\frac{q}{k-1})^j (1 - \frac{q}{k-1})^{d-j}.$$

Taking a union bound over $k-2$ labels, we have

$$Z_{i,1} \equiv \Pr(\cup_{k \neq i,0} N_{u,k} > C_k) \leq \sum_{k:k\neq i,0} \sum_{j=\lceil C_k \rceil}^{d} \binom{d}{j} (\frac{q}{k-1})^j (1 - \frac{q}{k-1})^{d-j}.$$

We define element v to be a two-hop-neighbor of u if there is at least one query that involves both the elements u and v. For the second type of errors we have to condition on the number of two-hop neighbors of u having label i for u when the true label is i. Let us denote the neighbor set of u by $\mathrm{Nbr}(u)$ and denote the number of nodes of label i in its neighbor set as $\mathrm{Nbr}_i(u)$. The probability of second type of error is

$$\begin{aligned} Z_{i,2} &\equiv \sum_{\hat{k}} \Pr(\mathrm{Nbr}_i(u) = \hat{k}) \Pr(N_{ui} < \lceil C_i \rceil \mid \mathrm{Nbr}_i(u) = \hat{k}) \\ &= \sum_{\hat{k}} \binom{d}{\hat{k}} p_i^{\hat{k}} (1-p_i)^{d-\hat{k}} \Pr(\hat{X_u} \neq X_u \mid \mathrm{Nbr}_i(u) = \hat{k}). \end{aligned}$$

Now we take two cases for the value of k. When $\hat{k} \leq \lfloor C_i \rfloor$ then

$$\begin{aligned} \Pr(N_{u,i} < \lceil C_i \rceil \mid \mathrm{Nbr}_i(u) = \hat{k}) = \sum_{i=0}^{\hat{k}} \binom{\hat{k}}{i} (\frac{q}{k-1})^i (1 - \frac{q}{k-1})^{\hat{k}-i} \\ \sum_{j=0}^{\lfloor C_i \rfloor - \hat{k} + i} \binom{d-\hat{k}}{j} (\frac{q}{k-1})^j (1 - \frac{q}{k-1})^{d-\hat{k}-j}, \end{aligned}$$

and when $\hat{k} \geq \lceil C_i \rceil$ then

$$\Pr(N_{u,i} < \lceil C_i \rceil \mid \mathrm{Nbr}_i(u) = \hat{k}) = \sum_{i=0}^{\lfloor C_i \rfloor} \binom{d-\hat{k}}{i} (\frac{q}{k-1})^i (1-\frac{q}{k-1})^{d-\hat{k}+i} \sum_{j=\hat{k}-\lfloor C_i \rfloor+i}^{\hat{k}} \binom{\hat{k}}{j} (\frac{q}{k-1})^j (1-\frac{q}{k-1})^{\hat{k}-j}.$$

Now, if the true label of u is 0

$$Z_0 = \cup_{k \neq 0} \Pr(N_{u,k} > C_k) = \sum_k \sum_{j=\lceil C_k \rceil}^{d} \binom{d}{j} (\frac{q}{k-1})^j (1-\frac{q}{k-1})^{d-j}.$$

The total probability of error is going to be

$$\delta = p_0 Z_0 + \sum_{i \neq 0} p_i (Z_{i,1} + Z_{i,2}).$$

Acknowledgements: This research is supported in parts by NSF Awards CCF-BSF 1618512, CCF 1642550 and an NSF CAREER Award CCF 1642658. The authors thank Barna Saha for many discussions on the topics of this paper. The authors also thank the volunteers who participated in the crowdsourcing experiments for this paper.

References

[1] E. Abbe, A. S. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *IEEE Trans. Information Theory*, 62(1):471–487, 2016.

[2] N. Alon and J. H. Spencer. *The probabilistic method*. John Wiley & Sons, 2004.

[3] H. Ashtiani, S. Kushagra, and S. Ben-David. Clustering with same-cluster queries. In *Advances In Neural Information Processing Systems*, pages 3216–3224, 2016.

[4] H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh. Are bitvectors optimal? *SIAM Journal on Computing*, 31(6):1723–1744, 2002.

[5] V. B. Chandar. *Sparse graph codes for compression, sensing, and secrecy*. PhD thesis, Massachusetts Institute of Technology, 2010.

[6] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[7] D. Firmani, B. Saha, and D. Srivastava. Online entity resolution using an oracle. *PVLDB*, 9(5):384–395, 2016.

[8] R. Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[9] A. Gruenheid, B. Nushi, T. Kraska, W. Gatterbauer, and D. Kossmann. Fault-tolerant entity resolution with the crowd. *CoRR*, abs/1512.00537, 2015.

[10] A. Gruenheid, B. Nushi, T. Kraska, W. Gatterbauer, and D. Kossmann. Fault-tolerant entity resolution with the crowd. *arXiv preprint arXiv:1512.00537*, 2015.

[11] B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming: Extensions. *IEEE Transactions on Information Theory*, 62(10):5918–5937, 2016.

[12] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.

[13] D. R. Karger, S. Oh, and D. Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research*, 62(1):1–24, 2014.

[14] F. Lahouti and B. Hassibi. Fundamental limits of budget-fidelity trade-off in label crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 5059–5067, 2016.

[15] Q. Liu, J. Peng, and A. T. Ihler. Variational inference for crowdsourcing. In *Advances in neural information processing systems*, pages 692–700, 2012.

[16] A. Makhdoumi, S.-L. Huang, M. Médard, and Y. Polyanskiy. On locally decodable source coding. In *Communications (ICC), 2015 IEEE International Conference on*, pages 4394–4399. IEEE, 2015.

[17] J. L. Massey. Joint source and channel coding. Technical report, DTIC Document, 1977.

[18] A. Mazumdar, V. Chandar, and G. W. Wornell. Update-efficiency and local repairability limits for capacity approaching codes. *IEEE Journal on Selected Areas in Communications*, 32(5):976–988, 2014.

[19] A. Mazumdar, V. Chandar, and G. W. Wornell. Local recovery in data compression for general sources. In *Information Theory (ISIT), 2015 IEEE International Symposium on*, pages 2984–2988. IEEE, 2015.

[20] A. Mazumdar and B. Saha. A theoretical analysis of first heuristics of crowdsourced entity resolution. *The Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, 2017.

[21] A. Mazumdar and B. Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems (NIPS) 31*, 2017.

[22] A. Mazumdar and B. Saha. Query complexity of clustering with side information. In *Advances in Neural Information Processing Systems (NIPS) 31*, 2017.

[23] G. Miller and D. Burshtein. Bounds on the maximum-likelihood decoding error probability of low-density parity-check codes. *IEEE Transactions on Information Theory*, 47(7):2696–2710, 2001.

[24] A. Montanari and E. Mossel. Smooth compression, gallager bound and nonlinear sparse-graph codes. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 2474–2478. IEEE, 2008.

[25] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[26] A. Pananjady and T. A. Courtade. Compressing sparse sequences under local decodability constraints. In *Information Theory (ISIT), 2015 IEEE International Symposium on*, pages 2979–2983. IEEE, 2015.

[27] M. Patrascu. Succincter. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 305–313. IEEE, 2008.

[28] D. Prelec, H. S. Seung, and J. McCoy. A solution to the single-question crowd wisdom problem. *Nature*, 541(7638):532–535, 2017.

[29] A. Vempaty, L. R. Varshney, and P. K. Varshney. Reliable crowdsourcing for multi-class labeling using coding theory. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):667–679, 2014.

[30] V. Verroios and H. Garcia-Molina. Entity resolution with crowd errors. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 219–230, 2015.

[31] N. Vesdapunt, K. Bellare, and N. Dalvi. Crowdsourcing algorithms for entity resolution. *PVLDB*, 7(12):1071–1082, 2014.

[32] R. K. Vinayak and B. Hassibi. Crowdsourced clustering: Querying edges vs triangles. In *Advances in Neural Information Processing Systems*, pages 1316–1324, 2016.

[33] E. Viola. Bit-probe lower bounds for succinct data structures. *SIAM Journal on Computing*, 41(6):1593–1604, 2012.

[34] J. Wang, T. Kraska, M. J. Franklin, and J. Feng. Crowder: Crowdsourcing entity resolution. *PVLDB*, 5(11):1483–1494, 2012.

[35] D. Zhou, S. Basu, Y. Mao, and J. C. Platt. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems*, pages 2195–2203, 2012.