Algorithms for Data Science: Lecture on Interactive Clustering

Barna Saha

1 Finding the Maximum Likelihood Clustering of V with faulty oracle

We can view the clustering problem as following. We have an undirected graph $G(V \equiv [n], E)$, such that G is a union of k disjoint cliques $G_i(V_i, E_i)$, $i = 1, \ldots, k$. The subsets $V_i \in [n]$ are unknown to us; they are called the clusters of V. The adjacency matrix of G is a block-diagonal matrix. Let us denote this matrix by $A = (a_{i,j})$.

Now suppose, each edge of G is erased independently with probability p, and at the same time each non-edge is replaced with an edge with probability p. Let the resultant adjacency matrix of the modified graph be $Z = (z_{i,j})$. The aim is to recover A from Z.

Lemma 1. *The maximum likelihood recovery is given by the following:*

$$\max_{S_\ell, \ell=1,\cdots: V=\sqcup_\ell S_\ell} \prod_\ell \prod_{i,j\in S_\ell, i\neq j} P_+(z_{i,j}) \prod_{r,t,r\neq t} \prod_{i\in S_r, j\in S_t} P_-(z_{i,j})$$

$$= \max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \prod_\ell \prod_{i,j\in S_\ell, i\neq j} \frac{P_+(z_{i,j})}{P_-(z_{i,j})} \prod_{i,j\in V, i\neq j} P_-(z_{i,j}).$$

where, $P_+(1) = 1-p, P_+(0) = p, P_-(1) = p, P_-(0) = 1-p$.

Hence, the ML recovery asks for,

$$\max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \sum_\ell \sum_{i,j\in S_\ell, i\neq j} \ln \frac{P_+(z_{i,j})}{P_-(z_{i,j})}.$$

Note that,

$$\ln \frac{P_+(0)}{P_-(0)} = -\ln \frac{P_+(1)}{P_-(1)} = \ln \frac{p}{1-p}.$$

Hence the ML estimation is,

$$\max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \sum_\ell \sum_{i,j\in S_\ell, i\neq j} \omega_{i,j}, \tag{1}$$

where $\omega_{i,j} = 2z_{i,j} - 1, i \neq j$, i.e., $\omega_{i,j} = 1$, when $z_{i,j} = 1$ and $\omega_{i,j} = -1$ when $z_{i,j} = 0, i \neq j$. Further $\omega_{i,i} = z_{i,i} = 0, i = 1, \ldots, n$.

Note that (1) is equivalent to finding correlation clustering in G with the objective of maximizing the consistency with the edge labels, that is we want to maximize the total number of positive intra-cluster edges and total number of negative inter-cluster edges [1, 3, 2]. This can be seen as follows.

$$\max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \sum_\ell \sum_{i,j\in S_\ell, i\neq j} \omega_{i,j}$$

$$\equiv \max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \Big[\sum_\ell \sum_{i,j\in S_\ell, i\neq j} \big|(i,j) : \omega_{i,j} = +1\big| - \big|(i,j) : \omega_{i,j} = -1\big|\Big] + \sum_{i,j\in V, i\neq j} \big|(i,j) : \omega_{i,j} = -1\big|$$

$$= \max_{S_\ell, \ell=1,\cdots: V=\sqcup_{\ell=1} S_\ell} \Big[\sum_{\ell} \sum_{i,j \in S_\ell, i \neq j} \big|(i,j) : \omega_{i,j} = +1\big| + \Big[\sum_{r,t: r \neq t} \big|(i,j) : i \in S_r, j \in S_t, \omega_{i,j} = -1\big| \Big].$$

Therefore (1) is same as correlation clustering. Also, note that, we have a random instance of correlation clustering here, and not a worst case instance.

1.1 Algorithm

We assume the number of clusters k is known. For unknown k, see [4].

Theorem 1. *There exists a polynomial time algorithm with query complexity* $\tilde{O}(\frac{nk^2}{(2p-1)^4})$ *for* Crowd-Cluster *with error probability* p*, which recovers all clusters of size at least* $\Omega(\frac{k \log n}{(2p-1)^4})$.

Algorithm 2. Let $N = \frac{64k^2 \log n}{(1-2p)^4}$. We define two thresholds $T(a) = pa + \frac{6}{(1-2p)}\sqrt{N \log n}$ and $\theta(a) = 2p(1-p)a + 2\sqrt{N \log n}$. The algorithm is as follows.

Phase 1-2C: Select a Small Subgraph. Initially we have an empty graph $G' = (V', E')$, and all vertices in V are unassigned to any cluster.

1. Select X new vertices arbitrarily from the unassigned vertices in $V \setminus V'$ and add them to V' such that the size of V' is N. If there are not enough vertices left in $V \setminus V'$, select all of them in X. Update $G' = (V', E')$ by querying for every (u, v) such that $u \in X$ and $v \in V'$ and assigning a weight of $\omega(u, v) = +1$ if the query answer is "yes" and $\omega(u, v) = -1$ otherwise .
2. Let $N^+(u)$ denote all the neighbors of u in G' connected by $+1$-weighted edges. We now cluster G'. Select every u and v such that $u \neq v$ and $|N^+(u)|, |N^+(v)| \geq T(|V'|)$. Then if $|N^+(u) \setminus N^+(v)| + |N^+(v) \setminus N^+(u)| \leq \theta(|V'|)$ (the symmetric difference of these neighborhoods) include u and v in the same cluster. Include in active all clusters formed in this step that have size at least $\frac{64k \log n}{(1-2p)^4}$. If there is no such cluster, abort. Remove all vertices in such clusters from V' and any edge incident on them from E'.

Phase 3C: Growing the Active Clusters.

1. For every unassigned vertex $v \in V \setminus V'$, and for every cluster $\mathcal{C} \in$ active, pick $\frac{16 \log n}{(1-2p)^2}$ distinct vertices, $u_1, u_2, ..., u_l$ in the cluster and query v with them. If the majority of these answers are "yes", then include v in $\mathcal{C}$.
2. Output all the clusters in active and move to Phase 1 step (1) to obtain the remaining clusters.

Analysis. Note that at every iteration, we consider a set of X new vertices from $V \setminus V'$ which have not been previously included in any cluster considered in active, and query all pairs in $X \times V' \setminus V$. Let A denote the fixed $n \times n$ matrix, where if $(i, j), i, j \in V$ is queried by the algorithm in any iteration, we include the query result there ($+1$ or -1), else the entry is empty which indicates that the pair was not queried by the entire run of the algorithm. This matrix A has the property that for any entry (i, j), if i and j belong to the same cluster and queried then $A(i, j) = +1$ with probability $(1 - p)$ and $A(i, j) = -1$ with probability p. On the other hand, if i and j belong to different clusters and queried then $A(i, j) = -1$ with probability $(1 - p)$ and $A(i, j) = +1$ with probability p. Note that the adjacency matrix of G' in any iteration is a submatrix of A which has no empty entry.

We first look at *Phase 1-2C*. At every iteration, our algorithm selects a submatrix of A corresponding to $V' \times V'$ after step 1. This submatrix of A has no empty entry. Let us call it A'. We show that if V' contains any subcluster of size $\geq \frac{64k \log n}{(2p-1)^4}$, it is retrieved by step 2 with probability at least $1 - \frac{1}{n^2}$. In that case, the iteration succeeds. Now the submatrices from one iteration to the other iteration can overlap, so we can only apply union bound to obtain the overall success probability, but that suffices. The probability that in step 2, the algorithm fails to retrieve any cluster of size at least $\frac{64k \log n}{(2p-1)^4}$ in any iteration is at most $\frac{1}{n^2}$. The

number of iterations is at most $k < n$, since in every iteration except possibly for the last one, V' contains at least one subcluster of that size by a simple pigeonhole principle. This is because in every iteration except possibly for the last one $|V'| = \frac{64k^2 \log n}{(2p-1)^4}$, and there are at most k clusters. Therefore, the probability that there exists at least one iteration which fails to retrieve the "large" clusters is at most $\frac{k}{n^2} \leq \frac{1}{n}$ by union bound. Thus all the iterations will be successful in retrieving the large clusters with probability at least $1 - \frac{1}{n}$.

Now, following the same argument as Lemma 3, each such cluster will be grown completely by Phase 3-C step (1), and will be output correctly in Phase 3-C step 2.

Lemma 2. *Let $c = \frac{64}{(1-2p)^4}$. Whenever G' contains a subcluster of size $ck \log n$, it is retrieved by Algorithm 2 in Phase 1-2C with high probability.*

Proof. Consider a particular iteration. Let $N^+(u)$ denote all the neighbors of u in G' connected by $+1$ edges. Let A' denote the corresponding submatrix of A corresponding to G'. We have $|V'| \leq N$ ($|V'| = N$ except possibly for the last iteration). Assume, $|V'| = N'$. Also $|V| = n$.

Let C_u denote the cluster containing u. We have

$$E[|N^+(u)|] = (1-p)|C_u| + p(N' - |C_u|) = pN' + (1-2p)|C_u|$$

Using the Chernoff Bound

$$\Pr(|N^+(u)| \in pN' + (1-2p)|C_u| \pm 2\sqrt{N \log n}) \geq 1 - \frac{1}{n^4}$$

Therefore for all u such that $|C_u| \geq \frac{8\sqrt{N \log n}}{(1-2p)^2}$, we have $|N^+(u)| > pN' + \frac{6}{(1-2p)}\sqrt{N \log n} = T(|V'|)$, and for all u such that $|C_u| \leq \frac{4\sqrt{N \log n}}{(1-2p)^2}$, we have $|N^+(u)| < pN' + \frac{6}{(1-2p)}\sqrt{N \log n}$ with probability at least $1 - \frac{1}{n^3}$ by union bound.

Consider all u such that $|N^+(u)| > T(|V'|)$. Then with probability at least $1 - \frac{1}{n^3}$, we have $|C_u| > \frac{4\sqrt{N \log n}}{(1-2p)^2}$. Let us call this set U. For every $u, v \in U, u \neq v$, the algorithm computes the symmetric difference of $N^+(u)$ and $N^+(v)$ which is

1. $2p(1-p)N'$ on expectation if u and v belong to the same cluster. And again applying the Chernoff bound, it is at most $2p(1-p)N' + 2\sqrt{N \log n}$ with probability at least $1 - \frac{1}{n^4}$.
2. $(p^2+(1-p)^2)(|C_u|+|C_v|)+2p(1-p)(N'-|C_u|-|C_v|) = 2p(1-p)N'+(1-2p)^2(|C_u|+|C_v|)$ on expectation if u and v belong to different clusters. Again using the Chernoff bound, it is at least $2p(1-p)N' + (1-2p)^2(|C_u| + |C_v|) - 2\sqrt{N \log n}$ with probability at least $1 - \frac{1}{n^4}$.

Therefore, for all u and v, either of the above two inequalities fail with probability at most $\frac{1}{n^2}$.

Now, since for all u if $|N^+(u)| > T(|V'|)$ then $|C_u| > \frac{4\sqrt{N \log n}}{(1-2p)^2}$ with probability $1 - \frac{1}{n^3}$, we get

for every u and v in U, if the symmetric difference of $N^+(u)$ and $N^+(v)$ is $\leq 2p(1-p)N' + 2\sqrt{N \log n} = \theta(|V'|)$, then u and v must belong to the same cluster with probability at least $1 - \frac{1}{n^2} - \frac{1}{n^3} \geq 1 - \frac{2}{n^2}$.

Hence, all subclusters of G' that have size at least $\frac{8\sqrt{N \log n}}{(1-2p)^2}$ will be retrieved correctly with probability at least $1 - \frac{2}{n^2}$. Now since $N' = N = \frac{64k^2 \log n}{(1-2p)^4}$ for all but possibly the last iteration, we have $\frac{8\sqrt{N \log n}}{(1-2p)^2} = \frac{64k \log n}{(1-2p)^4}$. Moreover, since there are at most k clusters in G and hence in G', there exists at least one subcluster of size $\frac{64k \log n}{(1-2p)^4}$ in G' in every iteration except possibly the last one, which will be retrieved.

Then, there could be at most $k < n$ iterations. The probability that in one iteration, the algorithm will fail to retrieve a large cluster by our analysis is at most $\frac{2}{n^2}$. Hence, by union bound over the iterations, the algorithm will successfully retrieve all clusters in Phase 1-2C with probability at least $1 - \frac{2}{n}$. □

Now, following the argument of Lemma 3, each subcluster of size $\frac{64k\log n}{(1-2p)^4}$ will be grown completely by Phase 3-C step (1).

Lemma 3. *The list* active *contains all the true clusters of V of size $\geq c'\log n$ at the end of the algorithm with high probability.*

Proof. From Lemma 2, any cluster that is added to active in Phase 2 is a subset of some original cluster in V with high probability, and has size at least $\frac{64k\log n}{(1-2p)^4}$. Moreover, whenever G' contains a subcluster of V of that size, it is retrieved by the algorithm and added to active.

When a vertex v is added to a cluster $\mathcal{C}$ in active, we have $|\mathcal{C}| \geq ck\log n$ at that time, and there exist $l = ck\log n$ distinct members of $\mathcal{C}$, say, $u_1, u_2, .., u_l$ such that majority of the queries of v with these vertices returned $+1$. Consider the situation that $v \notin \mathcal{C}$. Then the expected number of queries among the l queries that had an answer "yes" $(+1)$ is lp. We now use the following version of the Chernoff bound.

Lemma 4 (Chernoff Bound)**.** *Let $X_1, X_2, ..., X_n$ be independent binary random variables, and $X = \sum_{i=1}^{n} X_i$ with $E[X] = \mu$. Then for any $\epsilon > 0$*

$$\Pr[X \geq (1+\epsilon)\mu] \leq \exp\Big(-\frac{\epsilon^2}{2+\epsilon}\mu\Big)$$

and,

$$\Pr[X \leq (1-\epsilon)\mu] \leq \exp\Big(-\frac{\epsilon^2}{2}\mu\Big)$$

Hence, by the application of the Chernoff bound, $\Pr(v \text{ added to } \mathcal{C} \mid v \notin \mathcal{C}) \leq e^{-lp\frac{(\frac{1}{2p}-1)^2}{2+(\frac{1}{2p}-1)}} \leq \frac{1}{n^3}$.

On the other hand, if there exists a cluster $\mathcal{C} \in$ active such that $v \in \mathcal{C}$, then while growing $\mathcal{C}$, v will be added to $\mathcal{C}$ (either v already belongs to G', or is a newly considered vertex). This again follows by the Chernoff bound. Here the expected number of queries to be answered "yes" is $(1-p)l$. Hence the probability that less than $\frac{l}{2}$ queries will be answered yes is $\Pr(v \text{ not included in } \mathcal{C} \mid v \in \mathcal{C}) \leq \exp(-c\log n(1-p)\frac{(1-2p)^2}{8(1-p)^2}) \leq \exp(-\frac{2}{(1-p)}\log n) \leq \frac{1}{n^2}$. Therefore, for all v, if v is included in a cluster in active, the assignment is correct with probability at least $1-\frac{1}{n}$. Also, the assignment happens as soon as such a cluster is formed in active and v is explored (whichever happens first).

Furthermore, two clusters in active cannot be merged. Suppose, if possible there are two clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ which ought to be subset of the same cluster in V. Let without loss of generality $\mathcal{C}_2$ is added later in active. Consider the first vertex $v \in \mathcal{C}_2$ that is considered by our algorithm. If $\mathcal{C}_1$ is already there in active at that time, then with high probability v will be added to $\mathcal{C}_1$ in Phase 3. Therefore, $\mathcal{C}_1$ must have been added to active after v has been considered by our algorithm and added to G'. Now, at the time $\mathcal{C}_1$ is added to A in Phase 2, $v \in V'$, and again v will be added to $\mathcal{C}_1$ with high probability in Phase 2–thereby giving a contradiction.

This completes the proof of the lemma. □

Running time of the algorithm is dominated by the time required to run step 2 of Phase 1-2C. Computing trivially, finding the symmetric differences of $+1$ neighborhoods all $\binom{N}{2}$ pairs requires time $O(N^3)$. We can keep a sorted list of $+1$ neighbors of every vertex is $O(N^2\log n)$ time. Then, for every pair, it takes $O(N)$ time to find the symmetric difference. This can be reduced to $O(N^\omega)$ using fast matrix multiplication to compute set intersection where $\omega \leq 2.373$. Moreover, since each invocation of this step removes one cluster, there can be at most k calls to it and for every vertex, time required in Phase 3C over all the rounds is $O(\frac{k\log n}{(1-2p)^2})$. This gives an overall running time of $O(\frac{nk\log n}{(1-2p)^2} + kN^\omega) = O(\frac{nk\log n}{(1-2p)^2} + k^{1+2\omega}) = O(\frac{nk\log n}{(1-2p)^2} + k^{5.746})$. Without fast matrix multiplication, the running time is $O(\frac{nk\log n}{(1-2p)^2} + k^7)$.

The query complexity of the algorithm is $O(\frac{nk^2 \log n}{(2p-1)^4})$ since each vertex is involved in at most $O(\frac{k^2 \log n}{(2p-1)^4})$ queries within G' and $O(\frac{k \log n}{(2p-1)^2})$ across the active clusters. In fact, in each iteration, the number of queries within G' is $O(N^2)$ and since there could be at most k rounds, the overall query complexity is $O(\frac{nk \log n}{(2p-1)^2} + \min(\frac{nk^2 \log n}{(2p-1)^4}, kN^2))$. Thus we get Theorem 1.

References

[1] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[2] K. Makarychev, Y. Makarychev, and A. Vijayaraghavan. Correlation clustering with noisy partial information. In *Proceedings of The 28th Conference on Learning Theory*, pages 1321–1342, 2015.

[3] C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 712–728, 2010.

[4] A. Mazumdar and B. Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5790–5801, 2017.