Active Learning in the Geometric Block Model

Eli Chien,[1]* **Antonia Maria Tulino,**[2,3] **Jaime Llorca**[2]
[1]ECE, University of Illinois Urbana-Champaign, Illinois,
[2]Nokia Bell Labs, New Jersey, [3]DIETI, University of Naples Federico II, Italy
ichien3@illinois.edu, {a.tulino, jaime.llorca}@nokia-bell-labs.com

Abstract

The geometric block model is a recently proposed generative model for random graphs that is able to capture the inherent geometric properties of many community detection problems, providing more accurate characterizations of practical community structures compared with the popular stochastic block model. Galhotra et al. recently proposed a motif-counting algorithm for unsupervised community detection in the geometric block model that is proved to be near-optimal. They also characterized the regimes of the model parameters for which the proposed algorithm can achieve exact recovery. In this work, we initiate the study of active learning in the geometric block model. That is, we are interested in the problem of exactly recovering the community structure of random graphs following the geometric block model under arbitrary model parameters, by possibly querying the labels of a limited number of chosen nodes. We propose two active learning algorithms that combine the use of motif-counting with two different label query policies. Our main contribution is to show that sampling the labels of a vanishingly small fraction of nodes (sub-linear in the total number of nodes) is sufficient to achieve exact recovery in the regimes under which the state-of-the-art unsupervised method fails. We validate the superior performance of our algorithms via numerical simulations on both real and synthetic datasets.

1 Introduction

Community detection (or graph clustering) is one of the most important tasks in machine learning and data mining. In this problem, it is assumed that each node (or vertex) in a network (or graph) belongs to one of the underlying communities (or clusters), and that the topology of the network depends on these latent group memberships (or labels). The goal is to recover the communities by partitioning the nodes into different classes that match the labels up to a permutation. This problem has many applications, such as clustering in social networks (Fortunato 2010), detecting protein complexes in protein interaction networks (Chen and Yuan 2006), identifying customer interests in recommendation systems (Sahebi and Cohen 2011), and performing image classification and segmentation (Shi and Malik 2000).

The stochastic block model (SBM) is a popular random graph model for community detection that generalizes the well-known Erdös-Renyi model (Holland, Laskey, and Leinhardt 1983; Mossel, Neeman, and Sly 2015). In the SBM, the probability of having an edge between a pair of nodes depends only on the labels of the corresponding two nodes. In its simplest version, the SBM contains two communities of equal sizes, such that a pair of nodes from the same community are connected with probability p, and nodes from different communities are connected with probability q. Prior works (see (Abbe 2017) for an overview) have established the limits of unsupervised methods to achieve exact community detection (recovery) in terms of the relative difference between p and q.

However, many practical scenarios fall in the regimes where unsupervised methods fail to achieve exact recovery (difference between p and q below fundamental limit). It has then become apparent the need to understand if, in those regimes where unsupervised methods fail, we can still recover the correct community memberships by querying the labels of a small subset of nodes. The process of actively querying the labels of a subset of nodes, referred to as *active learning*, is a very useful tool for many machine learning applications where the acquisition of labeled data is expensive and/or time consuming (Cohn, Atlas, and Ladner 1994). In the active learning framework, we are allowed to query node labels up to a budget constraint in order to improve overall classification accuracy. The authors of (Gadde et al. 2016) showed that a sub-linear number of queries is sufficient to achieve exact recovery below the limit (in terms of difference between p and q) of unsupervised methods in the SBM, and that the number of queries needed for exact recovery depends on how far we are below such limit – hence providing a smooth trade-off between query complexity and clustering accuracy in the SBM.

While the SBM has gained a lot of popularity to benchmark the performance of clustering algorithms owing to its ease of tractability, it fails to capture very important properties of real networks, such as "transitivity" ('friends

*This work was done during Eli Chien's internship at Nokia Bell Labs, New Jersey.

having common friends') (Holland and Leinhardt 1971; Wasserman, Faust, and others 1994). Consider any three nodes in a graph, x, y, and z. Given the existence of edges between x and y, and between y and z, (partial) transitivity dictates that it is more likely than not that there also exists an edge between x and z. However, under the SBM, edges are assumed to exist independent of each other, conditioned on their respective node labels. Hence, in the SBM, the existence of edges (x, y) and (y, z) does not affect the probability of having edge (x, z), failing to capture transitivity.

In order to account for the apparent transitivity of many real networks, the authors of (Galhotra et al. 2018) proposed a random graph community detection model termed *geometric block model* (GBM). The GBM combines elements of the SBM with the well studied random geometric graph (RGG) model that has found important practical applications e.g., in wireless networking (Penrose and others 2003; Gupta and Kumar 1999; Devroye et al. 2011; Goel et al. 2005). In the GBM, the probability that and edge exists between two nodes depends, not only on the associated node labels, but also on their relative distance in the latent feature space. The authors in (Galhotra et al. 2018) experimentally validated the benefit of the GBM compared with the SBM to more accurately model real-world networks. In their follow-up work (Galhotra et al. 2019), they proposed a state-of-the-art near-optimal motif-counting algorithm that can achieve exact recovery with high probability when the GBM parameters are above the limit for exact unsupervised recovery. Interestingly, as we illustrate in Section 2.1, such limit is much higher than in the SBM, showing that clustering in the GBM is fundamentally harder than in the SBM, and hence that in many practical settings, unsupervised methods will not be sufficient to accurately cluster real-world networks.

1.1 Contributions

Motivated by the advantage of the GBM to more accurately characterize real-world networks and by the increased difficulty in clustering GBM-based networks (compared with the SBM), in this work, we initiate the study of active learning in the GBM.

We propose two active learning algorithms for the GBM that exactly recover the community memberships with high probability using a sub-linear number of queries, even in regimes below the limit of the state-of-the-art unsupervised algorithm in (Galhotra et al. 2019). Similar to the result of (Gadde et al. 2016) in the SBM, our results offer a smooth trade-off between query complexity and clustering accuracy in the GBM. Both algorithms exploit the idea of motif-counting to remove cross-cluster edges, while combining it with active learning in a different way. The first algorithm combines the use of motif-counting to remove cross-cluster edges (not necessarily all of them) with the minimax graph-based active learning algorithm S^2 (Dasarathy, Nowak, and Zhu 2015) to remove the remaining cross-cluster edges. The second algorithm employs a more aggressive version of motif-counting to remove *all* cross-cluster edges (possibly removing also some intra-cluster edges), and then queries for the label of at least one node from each disconnected component. Interestingly, our analysis of the motif-counting phase of our algorithms also leads to a slight improvement of the limit for exact unsupervised recovery derived in (Galhotra et al. 2019).

We test our algorithms extensively on both synthetic and real-world data. They improve the accuracy of the method in (Galhotra et al. 2018) from roughly 0.78 to 0.92 by querying no more than 4% of the nodes in two real-world datasets. We remark that this accuracy is much higher than that of the spectral method, which can only achieve roughly 0.6 accuracy on these same datasets. We also compare with the S^2 algorithm, which attains a slightly higher accuracy, but using at least 10 times more queries.

The full version of this paper, including the Supplement, can be found in (Chien, Tulino, and Llorca 2019).

1.2 Related work

Active learning on arbitrary graphs– Active learning on graphs has attracted significant attention in the recent research literature. Most previous works do not assume any knowledge of the underlying statistical model, and hence their performance guarantees depend on the parameters of the graph into consideration (Guillory and Bilmes 2009; Gu and Han 2012; Zhu, Lafferty, and Ghahramani 2003; Cesa-Bianchi et al. 2013; Dasarathy, Nowak, and Zhu 2015). While these approaches are fairly general, they tend be too pessimistic in settings where prior knowledge about the statistical model is available. In our work, we exploit the use of the minimax optimal graph-based active learning algorithm S^2 (Dasarathy, Nowak, and Zhu 2015) in combination with the prior knowledge of the underlying GBM.

Modeling transitivity– Prior attempts to include transitivity in random graph models include the Euclidean random graph (Sankararaman and Baccelli 2018), where edges between nodes are randomly and independently drawn as a function of the distance between the corresponding nodes' feature random variables. Differently from the GBM, clustering in this model requires, in addition to the graph, the values of the nodes' feature variables. Another transitivity driven model is the Gaussian mixture block model (Abbe et al. 2018), where node features are modeled via a Gaussian random vector with mean depending on the associated node label and identical variance. Two nodes are then connected by an edge if and only if their distance is smaller then some threshold. However, the authors of (Abbe et al. 2018) only use this model to empirically validate their proposed unsupervised clustering method. No theoretical results have yet been proved for this model.

Finally, we note that, while out of the scope of this paper, the use of hypergraphs provides another way to model transitivity, and that recent works have studied the generalization of the SBM in the hypergraph setting (Chien, Lin, and Wang 2018; 2019; Ghoshdastidar, Dukkipati, and others 2017; Ahn, Lee, and Suh 2016; Paul, Milenkovic, and Chen 2018).

2 Notation and the geometric block model

We use boldface upper case letters $\mathbf{A}$ to denote matrices and $[n]$ to denote the discrete set $\{1, 2, ..., n\}$. We use the standard asymptotic notation $f(n) = O(g(n))$ to denote

$\lim_{n\to\infty} |\frac{f(n)}{g(n)}| \leq C$ for some constant $C \geq 0$, and $f(n) = o(g(n))$ to denote $\lim_{n\to\infty} |\frac{f(n)}{g(n)}| = 0$.

We start by introducing the definition of the random geometric graph (RGG) model, which appeared as an alternative to the popular Erdös-Renyi graph.

Definition 2.1 (RGG, 2 dimensional torus case)**.** A random graph under $RGG(n, r)$ is a graph with n nodes, where each node $u \in [n]$ is associated with a latent feature vector $X_u \sim Unif[0, 1]$. Letting the distance between X_u and X_v be defined as $d_{uv} = \min(|X_u - X_v|, 1 - |X_u - X_v|)$, then, nodes u, v are connected by an edge under $RGG(n, r)$ if and only if $d_{uv} \leq r$.

Remark 2.1. Let $r \triangleq \theta\frac{\log(n)}{n}$ for some constant θ. It is well known that a random graph under $RGG(n, r)$ is connected with high probability if and only if $\theta > 1$ (Penrose and others 2003).

Next, we provide the definition of the GBM, which depends on the RGG in a similar manner as the SBM depends on the Erdös-Renyi graph.

Definition 2.2 (GBM, 2 dimensional torus case (Galhotra et al. 2018; 2019))**.** A random graph under GBM(n, σ, r_1, r_2) is a graph $G = (V, E)$ such that $V = [n]$ can be partitioned into two equal size components V_1 and V_2 determined by the label assignment σ. Specifically, $\sigma(i) = j$ if and only if $i \in V_j,\ \forall i \in [n], j = 1, 2$. Each node $u \in V$ is associated with a feature vector $X_u \sim Unif[0, 1]$ independently from each other. Letting the distance between X_u and X_v be defined as $d_{uv} = \min(|X_u - X_v|, 1 - |X_u - X_v|)$, then, $(u, v) \in E$ if and only if $d_{uv} \leq (r_1\mathbf{1}\{\sigma(u) = \sigma(v)\} + r_2\mathbf{1}\{\sigma(u) \neq \sigma(v)\})$, where $r_1 \geq r_2$ can depend on n.

Remark 2.2. Note that each cluster in GBM(n, σ, r_1, r_2) can be seen as an $RGG(n/2, r_1)$.

Remark 2.3. Let $(r_1, r_2) \triangleq (\theta_1, \theta_2)\frac{\log(n)}{n}$ for some constant $\theta_1 \geq \theta_2$. As shown in (Galhotra et al. 2018; 2019), if $\theta_1 - \theta_2 < 0.5$ or $\theta_1 < 1$, then no unsupervised method can correctly recover the community memberships with high probability. Since our focus is on the interesting $\frac{\log(n)}{n}$ scaling regime, with a slight abuse of notation, we will use GBM$(n, \sigma, \theta_1, \theta_2)$ to indicate $(r_1, r_2) = (\theta_1, \theta_2)\frac{\log(n)}{n}$ in the rest of the paper.

Note that in the GBM, a pair of nodes from the same community are connected with probability $2\theta_1\frac{\log(n)}{n}$, and nodes from different communities are connected with probability $2\theta_2\frac{\log(n)}{n}$. We refer to these two probabilities as the marginal distributions of the GBM.

2.1 Limits of unsupervised learning in the GBM and in the SBM

In this section, we compare the limits of unsupervised clustering on SBM and GBM by setting the marginal distributions of both models to be the same, and show how clustering in the GBM is fundamentally harder than in the SBM.

We first focus on the GBM. In order to achieve exact recovery, the algorithm of (Galhotra et al. 2019) requires the parameters of the GBM to satisfy certain sophisticated constraints. Due to space limitations, we only list Table 1 for some examples of GBM parameter values that satisfy such constraints. The complete description of the corresponding theorem is stated in the Supplement.

θ_2	1	2	3	4	5
min θ_1	8.96	12.63	15.9	18.98	21.93

Table 1: The minimum θ_1 for given θ_2 such that the algorithm in (Galhotra et al. 2019) would work.

We now turn to the SBM. Recall that in $SBM(n, \sigma, p, q)$ intra and inter community nodes are connected with probability p and q, respectively. Letting $p \triangleq a\frac{\log(n)}{n}, q \triangleq b\frac{\log(n)}{n}$, it is known that the state-of-the-art unsupervised method for the SBM requires $(\sqrt{a} - \sqrt{b})^2 \geq 2$ to achieve exact recovery. We set $b = 2\theta_2$ and $a = 2\theta_1$ to equate the marginal distributions to those of the GBM.

$\frac{b}{2}$	1	2	3	4	5
min $\frac{a}{2}$	4	5.83	7.46	9	10.47

Table 2: The minimum a for given b such that the best unsupervised method for SBM would work.

From Table 1 and 2, we can observe that exact recovery under the GBM requires much denser connections within clusters (large θ_1) than for the case of the SBM, implying that clustering under the GBM is much harder than under the SBM. This also means that many networks in practice, which are shown to follow the GBM more closely than the SBM (Galhotra et al. 2018), will likely fall in the regimes where unsupervised methods cannot achieve exact recovery, further motivating the importance of active learning for community detection in real-world networks that exhibit transitivity.

3 Active learning algorithms in the GBM

In what follows, we present two active learning algorithms for the GBM, whose pseudocode is described in Algorithms 1 and 2. Both algorithms are composed of two phases: a first unsupervised phase that builds on the motif-counting technique of (Galhotra et al. 2019) to remove cross-cluster edges, and a second phase that queries a subset of node labels until recovering the underlying clusters.

Phase 1 of Algorithm 1 removes as many cross-cluster edges as possible while preserving intra-cluster connectivity with high probability. During Phase 2, the S^2 algorithm is used to identify the remaining cross-cluster edges. In contrast, Algorithm 2 adopts a more aggressive edge removing policy during Phase 1. That is, it removes all cross-cluster edges with high probability. Note that in this case, intra-cluster connectivity may no be preserved. Nevertheless, during Phase 2, querying the label of one node in each disjoint component is sufficient to recover the underlying clusters.

One of the key elements of Phase 1 in the proposed algorithms is the motif-counting technique used in (Galhotra et al. 2019). Here, a motif is simply defined as a configuration of triplets (triangles) in the graph. For any edge (u, v), we count the number of triangles that cover edge (u, v). It is shown in (Galhotra et al. 2019) that this triangle count is statistically different depending on whether $\sigma(u) = \sigma(v)$ or $\sigma(u) \neq \sigma(v)$. More importantly, this count is also related to the feature distance d_{uv}. We will discuss this more precisely in Section 4.

Algorithm 1: Motif-counting with S^2

Input: Graph $G = (V, E)$, threshold E_T.
Output: Estimated labels $\hat{\sigma}$
Duplicate G by G_r
Phase 1:
for $(u, v) \in E$ **do**
  Calculate the number of triangles T^{uv} that cover the edge (u, v) on G, remove (u, v) from G_r if $T^{uv} \leq nE_T$.
end
Phase 2: Apply S^2 to G_r to get $\hat{\sigma}$. Terminate when we find 2 disjoint components.

Algorithm 2: Aggressive edge removing approach

Input: Graph $G = (V, E)$, parameter t_1.
Output: Estimated labels $\hat{\sigma}$
Duplicate G by G_r
Phase 1:
for $(u, v) \in E$ **do**
  Calculate the number of triangles T^{uv} that cover the edge (u, v) on G, remove (u, v) from G_r if $T^{uv} \leq (2\theta_2 + t_1)\log(n)$.
end
Phase 2:
Query one node for each disjoint components in G_r and assign labels according to queried nodes for each disjoint components.

In the following section, we show that under the assumption that $\theta_1 \geq 2\theta_2$[1] and $\theta_1 \geq 2$, both Algorithm 1 and Algorithm 2 guarantee exact recovery with sub-linear query complexity. However, note that if $\theta_1 < 2$, the underlying clusters may already contain disconnected components and, consequently, Algorithm 1 may not be able to preserve intra-cluster connectivity, requiring additional queries to achieve exact recovery. In this case, it is better to directly use Algorithm 2 even if exact recovery with sub-linear query complexity can no longer be guaranteed.

Finally, in the numerical results of Section 5, we show that under the assumption of perfect knowledge of the underlying GBM, Algorithm 2 has practically lower query complexity than Algorithm 1. However, when dealing with real datasets for which the parameters of the underlying GBM are not available, Algorithm 1 is shown to be more robust to the uncertainty of the GBM parameters.

[1]Note that the condition $\theta_1 \geq 2\theta_2$ is stronger than our model assumption $\theta_1 \geq \theta_2$

4 Analysis of algorithms

In this section, we provide theoretical guarantees for our algorithms, and sketch the associated proofs. Detailed proofs are deferred to the Supplement. We first state the result for the triangle count distribution.

Lemma 4.1 (Lemma 11 and Lemma 12 in (Galhotra et al. 2019))**.** Assume $\theta_1 \geq 2\theta_2$. Let $\mathbf{A}$ be the adjacency matrix of GBM$(n, \sigma, \theta_1, \theta_2)$. For any pair of nodes u, v with $A_{uv} = 1$, let $d_{uv} = x \triangleq \phi\frac{\log(n)}{n}$ and let the count of the triangles that cover edge (u, v) be $T^{uv}(x) \triangleq |\{z \in V : A_{uz} = A_{vz} = 1\}|$. If $\sigma(u) \neq \sigma(v)$, then

$$T^{uv}(x) \sim Bin(n-2, 2\theta_2\frac{\log(n)}{n}).$$

If $\sigma(u) = \sigma(v)$, then

$$T^{uv}(x) \sim Bin(\frac{n}{2}-2, (2\theta_1-\phi)\frac{\log(n)}{n}) + \mathbf{1}\{\phi \leq 2\theta_2\}Bin(\frac{n}{2}, (2\theta_2-\phi)\frac{\log(n)}{n}).$$

Lemma 4.1 shows that indeed the triangle count is an informative metric to distinguish the cases of $\sigma(u) = \sigma(v)$ and $\sigma(u) \neq \sigma(v)$. It is also strongly related to the distance between node features d_{uv}. See Figure 2a for visualization.

4.1 Analysis of Algorithm 1

In the following, we state the theoretical guarantees of Algorithm 1 under two different regimes using Theorems 4.2 and 4.3. To this end, we first define

$$t_1 = \inf\left\{t \geq 0 : (2\theta_2 + t)\log(\frac{2\theta_2 + t}{2\theta_2}) - t > 1\right\}. \quad (1)$$

Theorem 4.2. Under the assumption that $\theta_1 > 2\theta_2 \geq 2$, set

$$\eta = \inf\left\{t \geq 0 : (\theta_1 + \theta_2 - 2 - t)\log(\frac{\theta_1 + \theta_2 - 2 - t}{\theta_1 + \theta_2 - 2}) + t > 1\right\},$$
$$E_T = (\theta_1 + \theta_2 - 2 - \eta)\frac{\log(n)}{n}. \quad (2)$$

If $\theta_1 - \theta_2 - 2 - \eta > t_1$, then, after **Phase 1**, Algorithm 1 already recovers the communities up to a permutation with probability at least $1 - o(1)$. If $t_1 > \theta_1 - \theta_2 - 2 - \eta > 0$, then after **Phase 2**, Algorithm 1 recovers the communities up to a permutation with probability at least $1 - o(1)$, and query complexity at most

$$O(n^{1-\epsilon}\log(n)^3 + \log(n)), \quad (3)$$

where

$$\epsilon = (\theta_1+\theta_2-2-\eta)\log(\frac{\theta_1 + \theta_2 - 2 - \eta}{2\theta_2}) - (\theta_1-\theta_2-2-\eta).$$

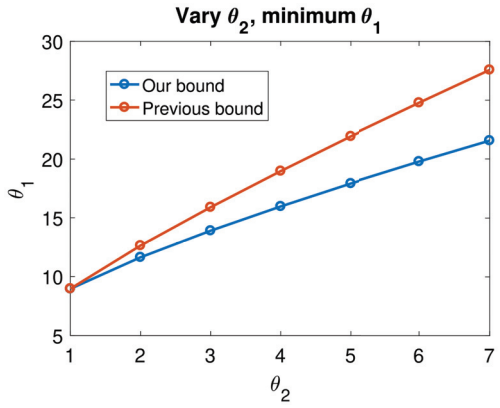


Figure 1: The minimum gap between θ_1 and θ_2 required for exact unsupervised clustering under Theorem 4.2 versus the state-of-the-art bound of (Galhotra et al. 2019).

Theorem 4.3. Under the assumption that $2\theta_2 \leq 2$, $\theta_1 \geq 2$, the same theoretical guarantees for Algorithm 1 stated in Theorem 4.2 can be derived by redefining

$$\eta = \inf\left\{t \geq 0 : (2\theta_1 - 2 - t)\log(\frac{2\theta_1 - 2 - t}{2\theta_1 - 2}) + t > 2\right\},$$
$$E_T = \frac{1}{2}(2\theta_1 - 2 - \eta)\frac{\log(n)}{n},$$
$$\epsilon = (\frac{1}{2}(2\theta_1 - 2 - \eta))\log(\frac{\frac{1}{2}(2\theta_1 - 2 - \eta)}{2\theta_2}) - (\frac{1}{2}(2\theta_1 - 2 - \eta) - 2\theta_2). \quad (4)$$

Remark 4.1. Note that for any fixed θ_2 such that $\theta_1 \geq 2\theta_2$ with $\theta_1 \geq 2$, $1 - \epsilon$ decays as θ_1 grows. Thus, Algorithm 1 provides a smooth trade-off between clustering accuracy and query complexity. Interestingly, Theorem 4.2 shows that when $\theta_1 - \theta_2 - 2 - \eta > t_1$, we can achieve exact recovery without any queries. We numerically show that this result gives an improvement over the previously known bound for unsupervised methods given in (Galhotra et al. 2019) for a wide range of θ_2 (See Figure 1).

In the following, we focus on proving Theorem 4.2, since Theorem 4.3 can be proved analogously. In order to prove Theorem 4.2, we will use the theoretical guarantee of the S^2 algorithm (Theorem 4.4) and two technical lemmas.

Theorem 4.4 (Simplified Theorem 3 in (Dasarathy, Nowak, and Zhu 2015))**.** Let C be the set of cross-cluster edges in graph G with latent labels σ. Let ∂C be the set of nodes associated with at least 1 cross-cluster edge. Suppose that each cluster is connected and has diameter at most D. If the S^2 algorithm uses at least

$$\frac{\log(2/\delta)}{\log(2)} + \lceil\log_2(n)\rceil + (\min(|\partial C|, |C|) - 1)(\lceil\log_2(2D + 1)\rceil + 1)$$

queries, then with probability at least $1-\delta$, the S^2 algorithm recovers the clusters exactly.

Remark 4.2. We note that while the original analysis in (Dasarathy, Nowak, and Zhu 2015) only uses the term $|\partial C|$ in the query complexity, the authors of (Chien, Zhou, and Li 2019) slightly improve it by including the term $\min(|\partial C|, |C|)$, which better serves our analysis.

Lemma 4.5. Assume $\theta_2 \geq 1$. Let

$$\eta = \inf\left\{t \geq 0 : (\theta_1 + \theta_2 - 2 - t)\log(\frac{\theta_1 + \theta_2 - 2 - t}{\theta_1 + \theta_2 - 2}) + t > 1\right\}.$$

Then, by choosing $E_T = (\theta_1 + \theta_2 - 2 - \eta)\frac{\log(n)}{n}$, **Phase 1** of Algorithm 1 is guaranteed to generate a graph G_r whose underlying communities are connected.

Lemma 4.6. Assume $\theta_2 \geq 1$ and set E_T as in Lemma 4.5. Let C be the set of cross-cluster edges in G_r. If $\theta_1 - \theta_2 - 2 - \eta > t_1$, then with probability at least $1 - o(1)$, we have $|C| = 0$. If $t_1 > \theta_1 - \theta_2 - 2 - \eta > 0$, we have

$$|C| \leq \frac{\theta_2}{2} n^{1-\epsilon}(\log(n))^2$$

with probability at least $1 - o(1)$, where

$$\epsilon = (\theta_1 + \theta_2 - 2 - \eta)\log(\frac{\theta_1 + \theta_2 - 2 - \eta}{2\theta_2}) - (\theta_1 - \theta_2 - 2 - \eta)$$

Remark 4.3. Note that while Lemma 4.5 characterizes the threshold in **Phase 1** of Algorithm 1 that guarantees removing the most cross-cluster edges while maintaining intra-cluster connectivity, Lemma 4.6 provides a bound on the number of remaining cross-cluster edges. Such bound, together with the result stated in Theorem 4.4, is one of the key ingredients in the evaluation of the query complexity bound of Algorithm 1. A graphical interpretation of the parameters t_1 and η, as well as of the key steps of the proof of Lemma 4.6 is provided in Figures 2a and 2b.

We are now ready to provide the proof of Theorem 4.2.

Proof. (Proof of Theorem 4.2) The first half of Theorem 4.2 directly follows from Lemma 4.5 and 4.6. Hence, in the following, we focus on the case $t_1 > \theta_1 - \theta_2 - 2 - \eta > 0$. From Lemma 4.5, we know that with probability at least $1 - o(1)$, the underlying clusters of graph G_r (the graph returned by **Phase 1**) are still connected among each other. Then, by Theorem 4.4, we know that with probability at least $1 - \delta$, using at most

$$\frac{\log(2/\delta)}{\log(2)} + \lceil\log_2(n)\rceil + (\min(|\partial C|, |C|) - 1)(\lceil\log_2(n+1)\rceil + 1)$$

queries, we can recover the communities. Finally, by Lemma 4.6, we know that with probability at least $1 - o(1)$, $\min(|\partial C|, |C|) \leq |C| \leq \frac{\theta_2}{2} n^{1-\epsilon}\log(n)^2$. Hence, by union bound over all error events, we know that with probability at least $1 - o(1)$, we can recover the communities with at most

$$O(n^{1-\epsilon}\log(n)^3 + \log(n))$$

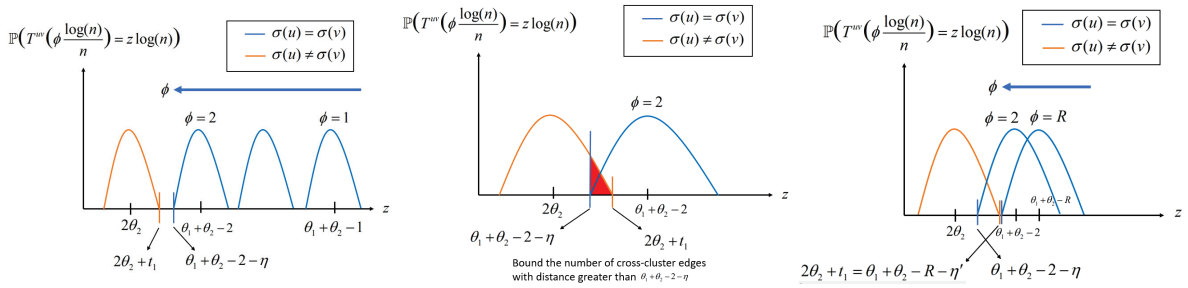
queries, by simply choosing $\delta = \frac{1}{n}$. □

4.2 Analysis of Algorithm 2

The next theorem provides the theoretical guarantee of Algorithm 2 under the assumption that $\theta_1 \geq 2\theta_2$, and $\theta_2 > 1$.

Theorem 4.7. Assume $\theta_1 \geq 2\theta_2$, $\theta_2 \geq 1$, and $2\theta_2 + t_1 > \theta_1 + \theta_2 - 2 - \eta$. With probability at least $1 - o(1)$, Algorithm 2 exactly recovers the underlying clusters with query complexity at most

$$\frac{3}{2} n^{1-R/2} + 2,$$



(a) $\theta_1 - \theta_2 - 2 - \eta > t_1$. (b) $\theta_1 - \theta_2 - 2 - \eta < t_1$. (c) $\theta_1 - \theta_2 - 2 - \eta < t_1$.

Figure 2: Figure (a) illustrates Lemma 4.1 and the intuition behind parameters t_1 and η. Figure (b) illustrates the main idea behind the proof of Lemma 4.6. We use the error probability (red area) to compute the expected number of cross-cluster edges that are kept after **Phase 1** of Algorithm 1. Then, we use Markov inequality to give the high probability bound for the number of cross-cluster edges in G_r. Figure (c) illustrates the idea behind parameter R in Lemma 4.8, which is chosen to be the largest number such that $T^{uv}(R\frac{\log(n)}{n}) \geq 2\theta_2 + t_1$ for intra-cluster edges (blue) with high probability.

where

$$R = \sup_{\min(\theta_1-\theta_2-t_1,2)>r>0} \left\{ (2\theta_2+t_1)\log(\frac{2\theta_2+t_1}{\theta_1+\theta_2-r}) + (\theta_1+\theta_2-r-(2\theta_2+t_1)) > 1 \right\}.$$

Note that if $2\theta_2 + t_1 < \theta_1 + \theta_2 - 2 - \eta$, since Algorithm 2 sets the threshold for the triangle count to $2\theta_2 + t_1$, it is immediate to note (see Figure 2a) that all cross-cluster edges will be removed while preserving all intra-cluster edges whose distance $\phi\frac{\log(n)}{n}$ is less than $2\frac{\log(n)}{n}$. In order to proof Theorem 4.7, we need the following lemmas.

Lemma 4.8. Assume $\theta_1 \geq 2\theta_2$, $\theta_2 \geq 1$ and $2\theta_2 + t_1 > \theta_1 + \theta_2 - 2 - \eta$. All intra-cluster edges with distance less than R will not be removed in G_r, where

$$R = \sup_{\min(\theta_1-\theta_2-t_1,2)>r>0} \left\{ (2\theta_2+t_1)\log(\frac{2\theta_2+t_1}{\theta_1+\theta_2-r}) + (\theta_1+\theta_2-r-(2\theta_2+t_1)) > 1 \right\}$$

Figure 2c provides a graphical illustration of R. The key idea is to find the largest R such that all intra-cluster edges with distance smaller than R will not be removed with high probability during **Phase 1** of Algorithm 2.

Next, we characterize the number of disjoint components created in each cluster by **Phase 1** of Algorithm 2. To this end (see Remark 2.2), we resort to the following lemma.

Lemma 4.9 (Modification of Theorem 8.1 in (Han and Makowski 2008))**.** Given a random geometric graph $RGG(n,\tau)$ with $2\tau < 1$, let $\mathcal{C}_{n,\tau}$ be the probability mass function of the number of disjoint components minus one of $RGG(n,\tau)$, and let Π_λ denote a Poisson distribution with parameter λ. Let $d_{TV}(\mu,\nu) \triangleq \frac{1}{2}\sum_{x=0}^{\infty}|\mu(x)-\nu(x)|$ be the total variation of the two probability mass functions μ and ν on $\mathbb{N}$. We then have

$$d_{TV}(\mathcal{C}_{n,\tau}, \Pi_{\lambda_n(\tau)}) \leq B_n(\tau),$$

where

$$\lambda_n(\tau) = n(1-\tau)^n,\ B_n(\tau) = n(1-\tau)^n - (n-1)(1-\frac{\tau}{1-\tau})^n.$$

The key idea of the proof of Theorem 8.1 in (Han and Makowski 2008) is observing that the number of disjoint components can be related to the indicator functions of the spacing of uniform random variables. Note that these indicator functions are nothing but properly correlated Bernoulli random variables which can be approximated by a Poisson random variable where the total variation can be bounded via Stein-Chen's method (Chen 1975). See more details in (Han and Makowski 2008), and the modification for our setting in the Supplement.

Lemma 4.10 (Poisson tail bound (Clément Canonne 2019))**.** Let $X \sim \Pi_\lambda$ be a Poisson random variable with parameter λ. For all $y > 0$,

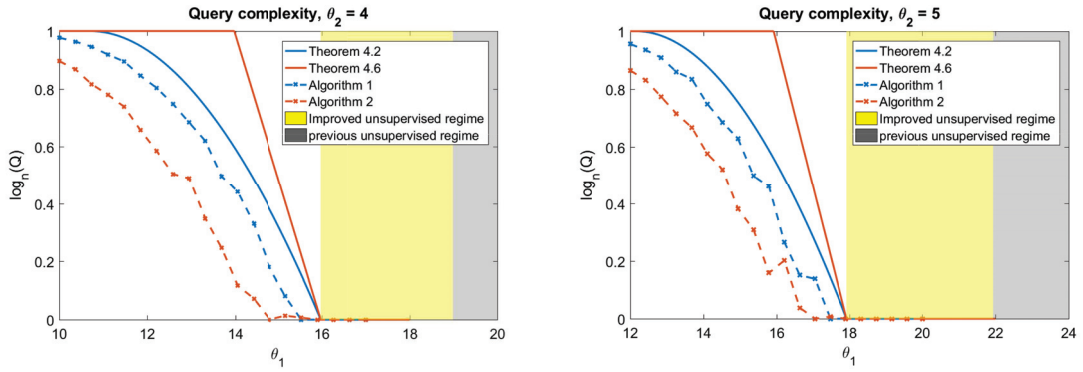
$$\mathbb{P}(X \geq \lambda + y) \leq \exp\left(-\frac{y^2}{2(\lambda+y)}\right).$$

We are now ready to state the sketch of the proof of Theorem 4.7. First, we use Lemma 4.8 to find the largest distance R such that, with high probability, all intra-cluster edges with distance smaller than R will not be removed during **Phase 1** of Algorithm 2. Next, we note that the number of disjoint components in G_r can be upper bounded by twice the number of disjoint components in an RGG$(\frac{n}{2}, R\frac{\log(n)}{n})$. Using Lemma 4.9, we approximate the number of disjoint components in RGG$(\frac{n}{2}, R\frac{\log(n)}{n})$ as a Poisson random variable with parameter given in Lemma 4.9. Combining Lemma 4.9 and 4.10, we are able to establish an upper bound on the number of disjoint components in G_r, which directly leads to the query complexity bound. The rigorous proof of Theorem 4.7 is deferred to the Supplement.

5 Experimental Results

5.1 Synthetic Datasets

We generate random graphs using a GBM$(n, \sigma, \theta_1, \theta_2)$ where $n = 1000$ and σ is chosen arbitrarily among the



(a) (b)

Figure 3: Query Complexity of our active learning algorithms in the GBM, where we use Q to denote the query complexity. Results are averaged over 20 independent trials. The light yellow shaded area indicates the improvement of our approach compared with (Galhotra et al. 2019) (grey shaded area) in the unsupervised setting. For Theorem 4.2 and 4.7, we only plot the main term in the theoretical bounds, $n^{1-\epsilon}$ and $n^{1-R/2}$.

equal-size community assignment. We plot the query complexity as a function of θ_1 for some fixed θ_2 in Figure 3. The figures for the other choices of θ_2 are deferred to the Supplement. Figure 3 plots the logarithm of the query complexity as a function of θ_1 for a given θ_2.[2] Note from the yellow shaded area that our results significantly improve the previously known bound for unsupervised clustering given in (Galhotra et al. 2019), and our theorems capture the behavior of the query complexity of Algorithms 1 and 2.

As expected, from Figure 3 we observe that for a fixed θ_2, as θ_1 decreases, we need more queries in order to achieve exact recovery. Note that, for a given θ_2, there is a large regime of θ_1 where unsupervised clustering fails, while our methods can achieve exact recovery with a small number of queries. For example, for $n = 10000$, $\theta_2 = 4$, and any value of θ_1 larger than around 13, we can achieve exact recovery with at most around 32 queries ($n^{0.5}$), which is just 3% of the total number of nodes.

Interestingly, while Theorem 4.2 provides a lower query complexity bound than Theorem 4.7, Figure 3 shows Algorithm 2 incurring lower query complexity in practice. This implies that our Theorem 4.7 can be tighter. In fact, while our current analysis only takes into account the edges preserved (not removed) with high probability, there may be other edges preserved only with constant probability. Each of these additional edges can potentially reduce the number of disjoint components by one. Nevertheless, this analysis is much more complicated since these edges are not independent, and is hence left for future work.

5.2 Real Datasets

- **Political Blogs (PB):** (Adamic and Glance 2005) It contains a list of political blogs from the 2004 US Election classified as liberal or conservative, and links between blogs. The clusters are of roughly the same size $(586, 636)$ with a total of 1222 nodes and 16714 edges.
- **LiveJournal (LJ):** (Yang and Leskovec 2015) The LiveJournal dataset is a free online blogging social network of around 4 million users. We extract the top two clusters of sizes (1430, 936) which consist of around 11.5K edges.

Experimental setting: For real-world networks, it is hard to obtain an exact threshold as the actual values of θ_1 and θ_2 are unknown. Hence, following the idea proposed in (Galhotra et al. 2018), we use a similar but much more intuitive approach compared with (Galhotra et al. 2018), which consists of 3 phases. In the first phase, we set a threshold T_1. We remove all edges $(u, v) \in E$ covered by less than T_1 triangles, and we identify $\mathcal{V}_0$ as the largest connected component in the resulting graph. In the second phase, we apply the S^2 algorithm on $\mathcal{V}_0$ and terminate it when we find 2 non-singleton disjoint components in $\mathcal{V}_0$. Finally, in the third phase, we take majority voting to decide the class of each node outside $\mathcal{V}_0$ based on the class of its already labeled neighbors. Note that, in contrast with the unsupervised method used in (Galhotra et al. 2018), where two more hyperparameters T_2 and T_3 are required, our active learning method only needs one hyperparameter T_1. We use GMPS18 to denote the unsupervised method in (Galhotra et al. 2018) and Spectral to denote the standard spectral method. All results are averaged over 100 independent trials.

Method	Accuracy		Query complexity (%)	
	PB	LJ	PB	LJ
Ours	0.931	0.912	3.7%	0.88%
Spectral	0.53	0.64	0	0
GMPS18	0.788	0.777	0	0
S^2	0.97	0.999	47.2 %	9.2%

Table 3: Performance on real-world datasets.

[2]Note that $\log_n(Q) < 1$ implies that a sub-linear number of queries can achieve exact recovery.

We choose $T_1 = 30$ for the PB dataset and $T_1 = 5$ for the LJ dataset. From Table 3, we can see that our active learning method only queries 3.7% of nodes and significantly improves the accuracy from 0.788 to 0.931 in the PB dataset. Also, note that if we directly apply S^2 without using triangle counting, it will query 47.2% of nodes before termination. Apparently, this is too expensive in terms of query complexity. A similar result can also be observed on the LJ dataset. Hence, integrating triangle counting is necessary for obtaining a practical solution in the active learning framework when we have a limited query budget.

Acknowledgments

The authors to thank Sainyam Galhotra for providing data and experiment details. This work was supported in part by NSF grant 1619129 and by grant 239 SBC PURDUE 4101-38050, NSF STC Center for Science of Information.

References

Abbe, E.; Boix, E.; Ralli, P.; and Sandon, C. 2018. Graph powering and spectral robustness. *arXiv preprint*.

Abbe, E. 2017. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research* 18(1):6446–6531.

Adamic, L. A., and Glance, N. 2005. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, 36–43. ACM.

Ahn, K.; Lee, K.; and Suh, C. 2016. Community recovery in hypergraphs. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 657–663. IEEE.

Cesa-Bianchi, N.; Gentile, C.; Vitale, F.; and Zappella, G. 2013. Active learning on trees and graphs. *arXiv preprint*.

Chen, J., and Yuan, B. 2006. Detecting functional modules in the yeast protein–protein interaction network. *Bioinformatics* 22(18):2283–2290.

Chen, L. H. 1975. Poisson approximation for dependent trials. *The Annals of Probability* 534–545.

Chien, I.; Lin, C.-Y.; and Wang, I.-H. 2018. Community detection in hypergraphs: Optimal statistical limit and efficient algorithms. In *International Conference on Artificial Intelligence and Statistics*, 871–879.

Chien, I.; Lin, C.-Y.; and Wang, I.-H. 2019. On the minimax misclassification ratio of hypergraph community detection. *IEEE Transactions on Information Theory*.

Chien, E.; Tulino, A. M.; and Llorca, J. 2019. Active learning in the geometric block model. *arXiv preprint*.

Chien, I. E.; Zhou, H.; and Li, P. 2019. HS^2: Active learning over hypergraphs with pointwise and pairwise queries. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 2466–2475.

Clément Canonne. 2019. A short note on poisson tail bounds.

Cohn, D.; Atlas, L.; and Ladner, R. 1994. Improving generalization with active learning. *Machine learning* 15(2):201–221.

Dasarathy, G.; Nowak, R.; and Zhu, X. 2015. S2: An efficient graph based active learning algorithm with application to nonparametric classification. In *Conference on Learning Theory*, 503–522.

Devroye, L.; György, A.; Lugosi, G.; Udina, F.; et al. 2011. High-dimensional random geometric graphs and their clique number. *Electronic Journal of Probability* 16:2481–2508.

Fortunato, S. 2010. Community detection in graphs. *Physics reports* 486(3-5):75–174.

Gadde, A.; Gad, E. E.; Avestimehr, S.; and Ortega, A. 2016. Active learning for community detection in stochastic block models. In *2016 IEEE International Symposium on Information Theory (ISIT)*, 1889–1893. IEEE.

Galhotra, S.; Mazumdar, A.; Pal, S.; and Saha, B. 2018. The geometric block model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Galhotra, S.; Mazumdar, A.; Pal, S.; and Saha, B. 2019. Connectivity in random annulus graphs and the geometric block model. *The International Conference on Randomization and Computation*.

Ghoshdastidar, D.; Dukkipati, A.; et al. 2017. Consistency of spectral hypergraph partitioning under planted partition model. *The Annals of Statistics* 45(1):289–315.

Goel, A.; Rai, S.; Krishnamachari, B.; et al. 2005. Monotone properties of random geometric graphs have sharp thresholds. *The Annals of Applied Probability* 15(4):2535–2552.

Gu, Q., and Han, J. 2012. Towards active learning on graphs: An error bound minimization approach. In *2012 IEEE 12th International Conference on Data Mining*, 882–887. IEEE.

Guillory, A., and Bilmes, J. A. 2009. Label selection on graphs. In *Advances in Neural Information Processing Systems*, 691–699.

Gupta, P., and Kumar, P. R. 1999. Critical power for asymptotic connectivity in wireless networks. In *Stochastic analysis, control, optimization and applications*. Springer. 547–566.

Han, G., and Makowski, A. M. 2008. Connectivity in one-dimensional geometric random graphs: Poisson approximations, zero-one laws and phase transitions.

Holland, P. W., and Leinhardt, S. 1971. Transitivity in structural models of small groups. *Comparative group studies* 2(2):107–124.

Holland, P. W.; Laskey, K. B.; and Leinhardt, S. 1983. Stochastic blockmodels: First steps. *Social networks* 5(2):109–137.

Mossel, E.; Neeman, J.; and Sly, A. 2015. Consistency thresholds for the planted bisection model. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, 69–75. ACM.

Paul, S.; Milenkovic, O.; and Chen, Y. 2018. Higher-order spectral clustering under superimposed stochastic block model. *arXiv preprint*.

Penrose, M., et al. 2003. *Random geometric graphs*, volume 5. Oxford university press.

Sahebi, S., and Cohen, W. 2011. Community-based recommendations: a solution to the cold start problem. In *Workshop on Recommender Systems and the Social Web (RSWEB)*.

Sankararaman, A., and Baccelli, F. 2018. Community detection on euclidean random graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2181–2200. SIAM.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *Departmental Papers (CIS)* 107.

Wasserman, S.; Faust, K.; et al. 1994. *Social network analysis: Methods and applications*, volume 8. Cambridge university press.

Yang, J., and Leskovec, J. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42(1):181–213.

Zhu, X.; Lafferty, J.; and Ghahramani, Z. 2003. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, volume 3.