arXiv:2504.19419v2 [cs.LG] 30 Oct 2025

Advancing Local Clustering on Graphs via Compressive Sensing: Semi-supervised and Unsupervised Methods

Zhaiming Shen
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332
zshen49@gatech.edu

Sung Ha Kang
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332
kang@math.gatech.edu

Abstract

Local clustering aims to identify specific substructures within a large graph without any additional structural information of the graph. These substructures are typically small compared to the overall graph, enabling the problem to be approached by finding a sparse solution to a linear system associated with the graph Laplacian. In this work, we first propose a method for identifying specific local clusters when very few labeled data are given, which we term semi-supervised local clustering. We then extend this approach to the unsupervised setting when no prior information on labels is available. The proposed methods involve randomly sampling the graph, applying diffusion through local cluster extraction, then examining the overlap among the results to find each cluster. We establish the co-membership conditions for any pair of nodes, and rigorously prove the correctness of our methods. Additionally, we conduct extensive experiments to demonstrate that the proposed methods achieve state of the art results in the low-label rates regime.

1 Introduction

The ability to learn from data by uncovering its underlying patterns and grouping it into distinct clusters based on latent similarities and differences is a central focus in machine learning. Over the past few decades, traditional clustering problems have been extensively studied as an unsupervised learning task, leading to the development of a wide range of foundational algorithms, such as k-means clustering [MacQueen, 1967], density-based clustering [Ester et al., 1996], spectral clustering [Ng et al., 2001, Zelnik-Manor and Perona, 2004], hierarchical clustering [Nielsen, 2016] and regularized k-means [Kang et al., 2011]. These foundational methods have, in turn, inspired numerous variants and adaptations tailored to specific data characteristics or application domains.

Researchers have also developed semi-supervised learning approach for clustering, which leverages both labeled and unlabeled data in various learning tasks. One of the most commonly used methods in graph-based semi-supervised learning is Laplace learning [Zhu et al., 2003]. Note that Laplacian learning sometimes is also called Label propagation [Zhu and Ghahramani, 2002], which seeks a graph harmonic function that extends the labels. Laplacian learning and its variants have been extensively applied in semi-supervised learning tasks [Zhou et al., 2004a,b, 2005, Ando and Zhang, 2007, Kang et al., 2014]. A key challenge with Laplacian learning type of algorithms is their poor performance in the low-label rates regime. To address this, recent approaches have explored p-Laplacian learning [El Alaoui et al., 2016, Flores et al., 2022], higher order Laplacian regularization [Zhou and Belkin, 2011], weighted nonlocal Laplacians [Shi et al., 2017], properly weighted Laplacian [Calder and Slepcev, 2019], and Poisson learning [Calder et al., 2020].

Preprint.

Those aforementioned clustering algorithms are all global clustering algorithms, recovering all cluster structures simultaneously. However, real-world applications often require identifying only specific substructures within large, complex networks. For example, in a social network, an individual may only be interested in connecting with others who share similar interests, while disregarding the rest of individuals. In such cases, global clustering methods become inefficient, as they generate excessive redundant information rather than focusing on the relevant local patterns.

In this paper, we turn our focus to a more flexible clustering method called *local clustering* or *local cluster extraction*. Local clustering focus only on finding one target cluster which contains those nodes of interest to us, and disregard the non-interest or background nodes. Researchers have investigated in this direction over the recent decades such as [Lang and Rao, 2004, Andersen et al., 2006, Kloster and Gleich, 2014, Spielman and Teng, 2013, Andersen et al., 2016, Veldt et al., 2019, Fountoulakis et al., 2020]. More recently, inspired by the idea of compressive sensing, Lai and Mckenzie [2020], Lai and Shen [2023], Shen et al. [2023] proposed a novel perspective for local clustering by finding the target cluster via solving a sparse solution to a linear system associated with the graph Laplacian matrix. Intuitively speaking, as local clustering only focus on finding "one cluster at a time", it is flexible in practice and can be more efficient and effective than global clustering algorithms.

Besides such merits that local clustering algorithms possesses, one big downside of it is the necessity of having the initial nodes (we call them seeds) of interest as prior knowledge, and it also requires a good estimate of the target cluster size. The seeds information is sometimes very limited and even unavailable, which makes local clustering algorithm less popular. We propose a clustering method which requires very few seeds (semi-supervised case) or no seeds (unsupervised case), see Figure 1 and 2 for illustration. We provide a detailed discussion of the procedure, with analysis on the correctness of the proposed method. The main contributions of our work are as follows:

1. We propose a semi-supervised (with very few labeled data) and an unsupervised (no labeled data) local clustering methods which outperform the state-of-the-arts local and non-local clustering methods in most cases.
2. For theoretical considerations, we relax the assumption on the spectral norm of perturbation on the graph Laplacian and prove the correctness of our method in the semi-supervised case. We also establish the co-membership conditions for any pair of nodes, and then prove the correctness of our method in the unsupervised case.
3. Computationally the proposed method can also show benefits that our semi-supervised method can find all the clusters simultaneously which improves the "one cluster at a time" feature of local clustering algorithms in terms of efficiency. We provide extensive experiments with comparisons to show the effectiveness of our methods in the low-label rates regime.

The rest of the paper is organized as follows. In Section 2, we present step-by-step procedure of the proposed methods in both semi-supervised and unsupervised settings, and prove their correctness under certain assumptions. In section 5, we show the experimental results of the proposed methods and compare them with the state-of-the-art semi-supervised clustering algorithms on various benchmark datasets. Finally, in Section 6, we conclude the paper, discuss its limitations and societal impact.

2 Model Assumptions

For a graph $G = (V, E)$, we use V to denote the set of all nodes, and E to denote the set of all edges. Suppose G has k non-overlapping underlying clusters $C_1, C_2, \cdots, C_k$, we use n_i to denote the size of C_i where $i = 1, 2, \cdots, k$, and use n to denote the total size of graph G. Further, we use A to denote the adjacency matrix (possibly weighted but non-negative) of graph G, and use D to denote the diagonal matrix whose diagonal entries is the degree of the corresponding vertex. We make the following assumptions on the graph model.

Assumption 1 *The non-zero entries in the diagonal block is denser than the non-zero entries in the off-diagonal block (after permutation according to the cluster membership). More precisely, we assume the graph Laplacian satisfies* $\|\Delta L\|_2 = \|L - L^{in}\|_2 = o(1)$ *and* $(\delta_n(L))^{\log n} = o(1)$ *as the graph size* $n \to \infty$. *The definition of RIP constant* δ_n *is provided in Definition 1 in Appendix D.*

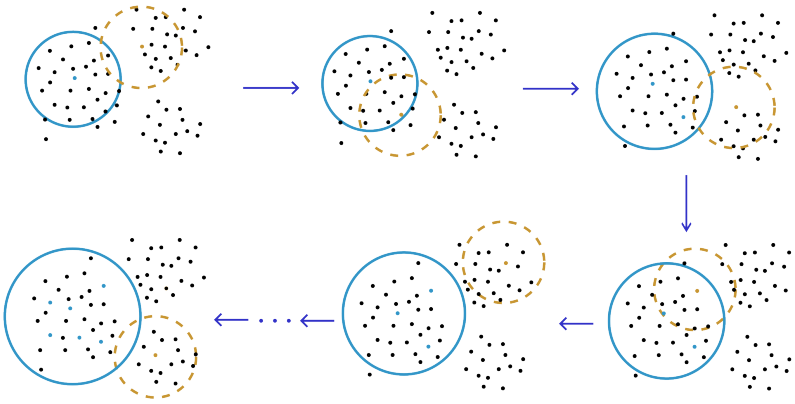


Figure 1: Illustration of semi-supervised local clustering (SSLC) for a single cluster. Each subplot indicates one iteration (Blue dots: seeds in Γ_1. Brown dots: randomly sampled node in each iteration).

Assumption 2 *The number of cluster $k = O(1) \geq 2$. The size of each cluster is not too small or too large, i.e., there exist $n_{\min} = \Theta(n)$ and $n_{\max} = \Theta(n)$ such that $n_{\min} \leq n_i \leq n_{\max}$ for any $i \geq 1$.*

Assumption 3 *For any distinct $i, j \in C_s$ with some $s \geq 1$, let $K_{i,j} \subset C_s$ be the set of nodes such that given any node in $K_{i,j}$ as the seed, LCE always finds node i and j into the same local cluster. We assume the cardinality of $K_{i,j}$ is at least $(1 - o(1))\frac{n_{\min}^2}{n}$ for any distinct pair of nodes i, j.*

Throughout the paper, all the notation $o(1)$ is with respect to the size of graph $n \to \infty$. Assumption 3 is essentially a type of homogeneity assumption on each cluster of the graph in order to guarantee the performance of LCE (see Algorithm 4) on a randomly chosen node.

One notable theoretical contribution in this paper is that we relax the assumption from $\|\Delta L\|_2 = o(n^{-1/2})$ (see Lemma 4 in Appendix C) [Shen et al., 2023] to $\|\Delta L\|_2 = o(1)$ (see Lemma 1 in Appendix A). We consider the local clustering tasks under two settings: semi-supervised case and unsupervised case.

3 Semi-supervised Local Clustering

For the semi-supervised case, we consider the following. Given a graph $G = (V, E)$ with underlying cluster $C_1, \cdots, C_k$ such that $V = \cup_{i=1}^k C_i$, $C_i \cap C_j = \emptyset$ for $1 \leq i, j \leq k$, $i \neq j$.

1. Suppose a very small portion of seeds $\Gamma_1 \subset C_1$ is available, we would like to find all the nodes in the target cluster C_1.
2. Suppose a very small portion of seeds $\Gamma_i \subset C_i$ is available for each $i = 1, \cdots, k$, we would like to assign all nodes to their corresponding underlying clusters.

Similar to the issue of Laplacian learning type of algorithms, local clustering approach such as LCE (see Algorithm 4) becomes less effective in the low-label rates regime. Therefore one wants to extract more seeds from each cluster before applying LCE. We illustrate the idea in Figure 1. We assume our target cluster is C_1, which is the cluster in the left of those three clusters illustrated in the first subplot of Figure 1.

The procedure of semi-supervised local clustering (SSLC) is summarized as follows:

1. Given a graph G, and initial seed(s) Γ_1 (indicated as the blue point in the first subplot), we first apply LCE to have a rough estimate $\tilde{C}_1$ (indicated by the solid blue circle) of C_1.
2. Randomly sample a node v (the brown point in the first subplot) in G and apply LCE to get a rough cluster around v (indicated by the dashed brown circle). Then check the overlap between solid blue and dashed brown circle. If the overlap contains the majority of the nodes in the brown circle, then add v into Γ_1, otherwise sample another node.

3. Continue this process and keep increasing the size of Γ_1 until a predetermined iteration number is reached (indicated in the process of from the second to the last subplots).

Note that this procedure can be applied when more than one cluster is of interest without increasing the number of iterations. The procedures for the single cluster case with Γ_1 and C_1 and multiple clusters case Γ_s and C_s are summarized in Algorithm 1 and 3 respectively. This method finds more seeds based on the initial seeds, and the user can flexibly determine the number of seeds wanted by increasing or decreasing the number of iterations. Theorem 1 essentially establishes the correctness of Algorithm 1 and 3.

Algorithm 1 Semi-supervised Local Clustering (SSLC) for a Single Cluster

Require: The adjacency matrix A of an underlying graph G, the initial seed(s) Γ_1 for the target cluster C_1, the estimated size n_1, the number of resampling iteration ℓ

Ensure: desired output cluster $C_1^{\#}$

$\tilde{C}_1 \leftarrow \mathrm{LCE}(A, n_1, \Gamma_1)$
for $i = 1 : \ell$ **do**
  $v \leftarrow$ uniform randomly sampled seed node from G
  $C^{\#} \leftarrow \mathrm{LCE}(A, n_1, v)$
  if $|\tilde{C}_1 \cap C^{\#}| > 0.5|C^{\#}|$ **then**
    $\Gamma_1 \leftarrow \Gamma_1 \cup \{v\}$
    $\tilde{C}_1 \leftarrow \mathrm{LCE}(A, n_1, \Gamma_1)$
  end if
end for
$C_1^{\#} \leftarrow \tilde{C}_1$

Theorem 1 *Suppose G satisfies Assumptions 1 - 3. Then when n (the size of G) gets large, for each $s = 1, \cdots, k$, we have*

$$\begin{cases} |\tilde{C}_s \cap C^{\#}| > (1 - o(1))|C^{\#}|, & \text{if } v \in C_s, \\ |\tilde{C}_s \cap C^{\#}| \leq o(|C^{\#}|), & \text{otherwise.} \end{cases}$$

Theorem 1 shows that whenever a node v being added to $\Gamma_1(\Gamma_s)$ based on the large overlap criterion, it satisfies $v \in C_1(C_s)$. This means that $\Gamma_1(\Gamma_s)$ only grow with the nodes within $C_1(C_s)$, which is the essential step to guarantee the correctness of Algorithm 1 and Algorithm 3.

4 Unsupervised Local Clustering

For unsupervised case, we assume the following: Given a graph $G = (V, E)$ with underlying cluster $C_1, C_2, \cdots$, where $V = \cup_{i \geq 1} C_i$, $C_i \cap C_j = \emptyset$ for all $i \neq j$. Assume neither the prior information about seeds nor the number of cluster k is given, the goal is to assign all the nodes to their corresponding underlying clusters.

In this case, we randomly sample and build a local cluster from the sampled node every time. We check its overlap with the local cluster obtained from any newly sampled node. After a certain number of iterations, the nodes from the same underlying cluster are more likely to be clustered together. We illustrate the idea of unsupervised local clustering in Figure 2. The procedure of unsupervised local clustering (USLC) is summarized as follows:

1. Given a graph G, in each iteration, we randomly sample a node and find its local cluster via LCE (as shown in the top row of Figure 2).
2. Based on the found local cluster, build the co-membership matrix in such a way that it outputs 1 in the location (i, j) when both i, j are contained in that found cluster, and outputs 0 otherwise.
3. Aggregate the clustering results from all iteration into a co-membership matrix M (roughly speaking, each entry $M_{i,j}$ represents the probability of a pair of nodes being clustered into the same local cluster). The values in each block of matrix M asymptotically converges to the same value, as the iteration number goes up (as shown in the bottom row of Figure 2).

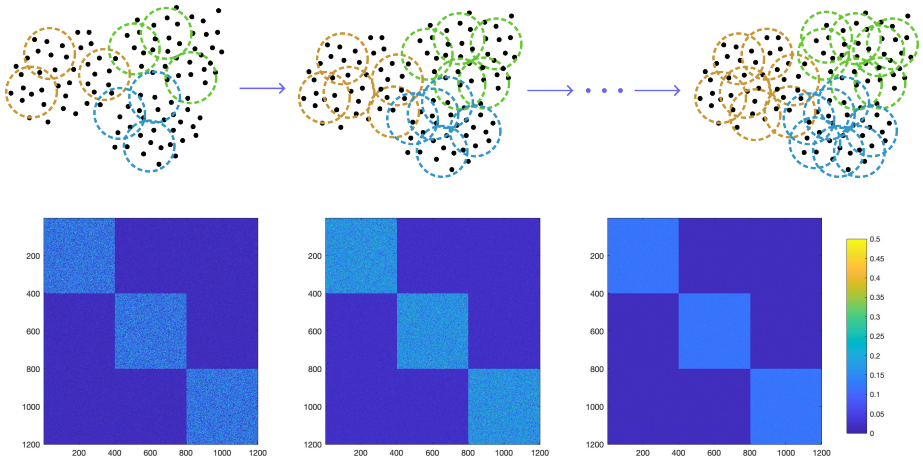


Figure 2: Illustration of USLC procedure. Top row: each dashed circle represents one iteration of LCE generated from a randomly sampled node, different colors indicate local clusters generated from nodes of different underlying clusters. Bottom row: Aggregated co-membership matrix.

4. Randomly sample a node i, and use a prescribed cutoff number δ to select all the nodes u such that $M_{i,j} > \delta$ and assign those j's as one of the clusters $C^{\#}$.
5. Delete the subgraph generated by $C^{\#}$ from G. Then keep iterating until the prescribed maximum number of clusters is reached or the remaining size of graph is too small.

We summarize its procedure in Algorithm 2. We first show in Proposition 1 that $M_{i,j}$, the probability of a pair of nodes coming from the same underlying cluster, is significantly different from the probability of a pair of nodes coming from two different underlying clusters. Then we show in Theorem 2 the correctness of Algorithm 2. The detailed proofs are deferred to Appendix A.

Algorithm 2 Unsupervised Local Clustering (USLC)

Require: The adjacency matrix A of an underlying graph G, the number of resampling iteration ℓ
Ensure: The desired clusters $C_1^{\#}, C_2^{\#}, \cdots$

initialize $s \leftarrow 1$
while $|G| < n_{\min}$ **do**
  initialize the comembership matrix M as zero matrix
  for $i = 1 : \ell$ **do**
    $v \leftarrow$ uniform randomly sampled seed node from G
    $C^{\#} \leftarrow \mathrm{LCE}(A, n_{\min}, v)$
    $M \leftarrow M + \frac{1}{\ell} \cdot \mathrm{comembership}(\mathbf{1}_{C^{\#}})$
  end for
  select a node v uniformly random such that $M_{v,u} > \delta$ for some $u \neq v$, with some appropriate $\delta \in (0,1)$, then find all the j such that $C_s^{\#} := \{j : M_{v,j} > \delta\}$
  Let $G^{(s)}$ be the subgraph spanned by $C_s^{\#}$
  $G \leftarrow G \setminus G^{(s)}$
  $s \leftarrow s + 1$
end while

It is worthwhile to mention that the value $n_{\min}$ can be a very rough lower bound, and Algorithm 2 is robust with respect to any reasonable choice of $n_{\min}$. Since the choice of δ for thresholding depends on $n_{\min}$, as long as we choose δ accordingly, the algorithm is robust. As shown in the proof Theorem 2 in Appendix A, we can use $\delta = \frac{1}{2}(\frac{n_{\min}}{n})^2$ as a general guideline in practice.

Proposition 1 *Suppose G satisfies Assumptions 1 - 3. Then, as n gets large, the co-membership matrix M obtained from Algorithm 2 has a clear block diagonal form (after permutation) as $L \to \infty$.*

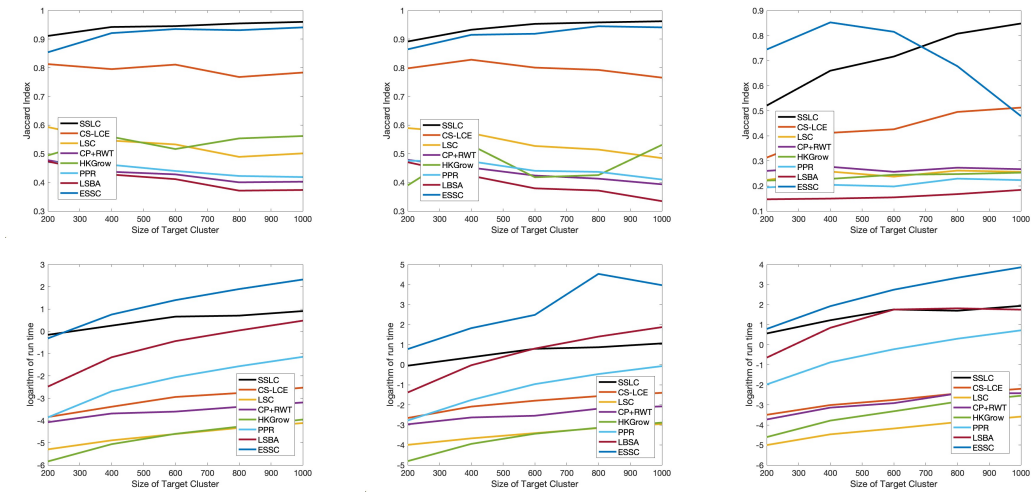


Figure 3: Plots of Jaccard index (over 100 trials) and logarithm of running time of SSLC for stochastic block model. (a) The first column shows experiment with three equal cluster size and single cluster C_1 extraction. (b) The middle column shows experiment with three equal cluster size and all clusters C_s extraction. (c) The right column shows experiment with unequal cluster sizes, focusing on the most dominent cluster.

More precisely, the entries in M satisfy

$$\begin{cases} M_{i,j} \geq (1-o(1))\frac{n_{\min}}{n}, & \text{for any } i=j, \\ M_{i,j} \geq (1-o(1))\frac{n_{\min}^2}{n^2}, & \text{for } i,j \in C_s, i \neq j, s \geq 1, \\ M_{i,j} \leq o\left(\frac{4n_{\min}^2(1-o(1))}{n^2}\right), & \text{otherwise.} \end{cases}$$

Theorem 2 *Suppose the assumptions in Theorem 1 hold. For any $v \in C_s, s \geq 1$, there exists some $\delta \in (0,1)$ and $C_s^{\#} = \{i : M_{v,i} > \delta\}$ such that $C_s^{\#} = C_s$. Consequently, Algorithm 2 assigns all nodes into their clusters correctly.*

It is worth of noting that Algorithms 1, 2 and 3 are applicable if there are outlier nodes, i.e., nodes which do not belong to any underlying clusters, presented in the graph. In such case, we extract all the clusters, and the remaining unclustered nodes are automatically classified as outlier nodes.

5 Experiments

We evaluate the performance of SSLC and USLC on both synthetic and real datasets, with a particular focus on clustering in the low-label rates regime (with label rates at most 1%). Additionally, we demonstrate the robustness of our methods by manually introducing outliers into the dataset and assessing their impact. All experiments are conducted on a local machine with 8-core Ryzen 7 7700X CPUs and 24 GB of RAM capacity. The runtime of SSLC/USLC is primarily determined by the LCE procedure, which is $O(nd_{\max}\log(n))$. In the regime $d_{\max} = O(\log n)$, this becomes $O(n\log^2 n)$. Therefore, if there are total k number of clusters, the total run time is $O(kn\log^2 n)$. Samples of demo code is available at `https://anonymous.4open.science/r/Iterative_LCE-78C4`.

Synthetic Data We first conduct two experiments for semi-supervised setting on stochastic block model (SBM) with different cluster sizes and different connection probabilities in Figure 3. The first cases (a) and (b) are symmetric SBM with three equal cluster size, the second case (c) is general SBM with three unequal clusters sizes. For the symmetric case, we consider both the single cluster extraction in (a) and all clusters extraction in (b). For the nonsymmetric case in (c), we only focus on extracting the most dominant cluster, i.e., the cluster with the largest connection probability. The parameters for generating data in both cases are presented in Appendix E. We compare against several other local clustering algorithms such as CS-LCE or LCE [Shen et al., 2023], LSC [Lai and Shen,

Table 1: Average accuracy and standard deviation on geometric data over 100 trials

Datasets	3 Lines	3 Circles	3 Moons
CP+RWT	82.1 (9.1)	96.1 (5.1)	85.4 (1.3)
LSC	89.0 (5.5)	96.2 (3.7)	85.3 (1.9)
LCE	92.4 (8.1)	97.6 (4.7)	96.8 (0.9)
SSLC	**94.8 (7.2)**	**98.2 (4.1)**	**97.3 (1.2)**

Table 2: Average accuracy and standard deviation on FashionMNIST over 30 trials

# Labels per class	0	1	2	3	4	5
Laplace	-	18.4 (7.3)	32.5 (8.2)	44.0 (8.6)	52.2 (6.2)	57.9 (6.7)
Random Walk	-	49.0 (4.4)	55.6 (3.8)	59.4 (3.0)	61.6 (2.5)	63.4 (2.5)
VolumeMBO	-	54.7 (5.2)	61.7 (4.4)	66.1 (3.3)	68.5 (2.8)	70.1 (2.8)
WNLL	-	44.6 (7.1)	59.1 (4.7)	64.7 (3.5)	67.4 (3.3)	70.0 (2.8)
p-Laplace	-	54.6 (4.0)	57.4 (3.8)	65.4 (2.8)	68.0 (2.9)	68.4 (0.5)
PoissonMBO	-	62.0 (5.7)	67.2 (4.8)	70.4 (2.9)	72.1 (2.5)	73.1 (2.7)
CutSSL	-	62.0 (5.7)	67.2 (4.8)	70.4 (2.9)	72.1 (2.5)	73.1 (2.7)
SSLC/USLC	**61.6 (6.2)**	**65.8 (4.1)**	**69.1 (3.2)**	**74.3 (2.3)**	**77.2 (2.6)**	**78.7 (2.3)**

2023], CP+RWT [Lai and Mckenzie, 2020], HKGrow [Kloster and Gleich, 2014], PPR [Andersen et al., 2006], ESSC [Wilson et al., 2014], and LBSA [Shi et al., 2019].

For the experiments on SBM, we set the number of initial seeds to be 1. We observe that SSLC has a better Jaccard index compared to other methods in most cases. Moreover, for symmetric model, the Jaccard index in both single cluster extraction and all clusters extraction are similar, while the running time of all clusters extraction is more advantageous towards SSLC (see also Remark 2).

To further validate that our assumption $\|\Delta L\|_2 = \|L - L^{in}\|_2 = o(1)$, we conduct experiments on symmetric stochastic block model by fixing the number of nodes, number of clusters, intra-connection probability, while varying the inter-connection probability. The results are provided in Table 5. We observe that the $\|\Delta L\|_2$ decreases as n increases, which is as desired.

Our experimental results are also connected to the theoretical results in the SBM literature. Note that when the signal-to-noise ratio $\mathrm{SNR} = \frac{(a-b)^2}{k(a+(k-1)b)} > 1$ (Kestem-Stigum threshold) in the SBM literature gives a theoretical bound for weak recovery of SBM in the sparse regime (with connectivity $\frac{a}{n}$ and $\frac{b}{n}$). For our exact recovery case (with connectivity $\frac{a\log n}{n}$ and $\frac{b\log n}{n}$), if we fix a, b, k and increase n, then $\|\Delta L\|$ decreases (illustrated in Table 5). In our experiments, we set $k = 3, p = \frac{a\log n}{n}, q = \frac{b\log n}{n}$ with $a = 6$ and $b = 1$, then we roughly satisfies the condition $\sqrt{a} - \sqrt{b} \approx \sqrt{k}$ for exact recovery, which is given in Theorem 14 and Remark 16 in Abbe [2018].

Besides SBM, we compare our algorithm against other compressive sensing based local clustering methods on geometric dataset consisting of different shapes (three lines, three circles, three moons). The 2D projection and more details of this dataset are provided in Appendix E.3. Note that spectral clustering often fails on these datasets when the noise level is large. However, compressive sensing based approaches work better. We use 1 label per class, the clustering results are shown in Table 1. We see that SSLC outperforms other compressive sensing based local clustering methods significantly.

Real Data For real datasets, we focus on clustering in the low-label rates regime on Fashion-MNIST [Xiao et al., 2017], CIFAR-10 [Krizhevsky et al., 2009], and Planetoid (Cora, CiteSeer, PubMed) [Yang et al., 2016]. To apply graph-based clustering algorithms on images, we first construct an auxiliary K-NN graph, and compute the pairwise distance from Gaussian kernel based on the Euclidean distance of the latent features with some scaling factors. Similar constructions have also appeared among others [Zelnik-Manor and Perona, 2004, Jacobs et al., 2018, Calder et al., 2020]. More detailed construction is provided in Appendix E.

We compare against other modern semi-supervised methods such as Laplace learning [Zhu et al., 2003], lazy random walks [Zhou and Scholkopf, 2004, Zhou et al., 2004a], weighted nonlocal

Table 3: Average accuracy and standard deviation on CIFAR-10 over 30 trials

# Labels per class	0	1	2	3	4	5
Laplace	-	10.4 (1.3)	11.0 (2.1)	11.6 (2.7)	12.9 (3.9)	14.1 (5.0)
Random Walk	-	36.4 (4.9)	42.0 (4.4)	45.1 (3.3)	47.5 (2.9)	49.0 (2.6)
VolumeMBO	-	38.0 (7.2)	46.4 (7.2)	50.1 (5.7)	53.3 (4.4)	55.3 (3.8)
WNLL	-	16.6 (5.2)	26.2 (6.8)	33.2 (7.0)	39.0 (6.2)	44.0 (5.5)
p-Laplace	-	26.0 (6.7)	35.0 (5.4)	42.1 (3.1)	48.1 (2.6)	49.7 (3.8)
PoissonMBO	-	41.8 (6.5)	50.2 (6.0)	53.5 (4.4)	56.5 (3.5)	57.9 (3.2)
CutSSL	-	41.8 (6.5)	50.2 (6.0)	53.5 (4.4)	56.5 (3.5)	57.9 (3.2)
SSLC/USLC	**48.2 (6.1)**	**51.2 (5.2)**	**57.1 (4.9)**	**59.7 (4.7)**	**61.3 (3.4)**	**63.5 (2.7)**

Table 4: Average accuracy and standard deviation of SSLC/USLC by adding 10% number of outliers

# Labels Per Class	0	1	2	3	4	5
FashionMNIST	55.6 (6.7)	60.4 (6.2)	65.6 (5.9)	69.9 (4.7)	73.5 (4.0)	74.8 (3.9)
CIFAR-10	45.1 (7.9)	47.3 (7.3)	51.6 (7.0)	55.3 (5.9)	58.1 (5.4)	60.1 (4.7)

laplacian (WNLL) [Shi et al., 2017], VolumeMBO [Jacobs et al., 2018], PoissonMBO [Calder et al., 2020], p-Laplace learning [Flores et al., 2022], and CutSSL [Holtz et al., 2024]. The results are summarized in Table 2 for FashionMNIST dataset, and Table 3 for CIFAR-10 dataset. Additional experimental results on Planetoid are included in Table 6 in Appendix F. Note that all the other methods can only handle the semi-supervised case, i.e., the number of label(s) per class is at least 1.

It is worthwhile to note that deep clustering method such as SimCLR [Chen et al., 2020] gives a strong baseline with 65% accuracy on FashionMNIST and 50% accuracy on CIFAR-10, which does not show clear advantage over our method, yet our method is much efficient and easy to implement.

To further demonstrate the robustness of our methods against outliers in the graph, we manually add some images consisting of random noise as outliers into datasets FashionMNIST and CIFAR-10. Specifically, we add the number of outliers equal to 10% of the original dataset size. Each outlier image is generated by setting its pixel values equal to the standard Gaussian random noise. For example, as shown in Figure 5, the last block consists entirely of outliers, exhibiting no community structure, while other blocks do. Such case is also called tight clustering in the literature [Tseng and Wong, 2005, Deng et al., 2024]. By comparing Table 4 with Table 2 and 3, it further confirms that the performance of our proposed methods remains largely unaffected in the presence of these outliers.

6 Conclusion and Discussion

Conclusion. We proposed a unified framework for extracting local clusters in both semi-supervised and unsupervised settings with theoretical guarantees. The methods require minimal label information in the semi-supervised case and no label information in the unsupervised case, achieving state-of-the-art results. Notably, the methods are very effective particularly in the low-label rates regime, and remain robust when outliers are presented. We point out two limitations of this work. First, our methods mostly work well for sparse graph. This is mainly due to sparse graph has clear clustering structure, meaning that it often have tightly connected clusters with fewer inter-cluster edges, making local algorithms efficient at detecting boundaries. Second, when the label rate is not low, our proposed method in the semi-supervised case do not exhibit advantages over other methods.

Broader Impact. While local clustering algorithms offer valuable insights for applications like community detection and recommendation systems, their potential misuse raises ethical concerns, particularly in contexts like surveillance or discriminatory profiling. By identifying tightly connected groups in social networks, these techniques can inadvertently reinforce biases (e.g., via homophily) or enable repression when deployed without transparency or consent. The societal impact hinges on intent—clustering can empower communities when used responsibly, but it risks harming marginalized groups if applied for unchecked monitoring or targeting. Practitioners should weigh considerations like purpose, bias mitigation, and user awareness to ensure ethical deployment, acknowledging that even mathematically neutral tools carry real-world consequences.

References

Emmanuel Abbe. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018.

Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. In *IEEE Symposium on Foundations of Computer Science*, pages 475–486. IEEE, 2006.

Reid Andersen, Shayan Oveis Gharan, Yuval Peres, and Luca Trevisan. Almost optimal local graph clustering using evolving sets. *Journal of the ACM (JACM)*, 63(2):1–31, 2016. doi: 10.1145/2856033. Article No. 15.

R. K. Ando and T. Zhang. Learning on graph with laplacian regularization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 25–32. MIT Press, 2007.

Thomas Blumensath and Mike E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

J. Calder and D. Slepcev. Properly-weighted graph laplacian for semi-supervised learning. *Applied Mathematics & Optimization*, Dec 2019.

Jeff Calder, Brendan Cook, Matthew Thorpe, and Dejan Slepcev. Poisson learning: Graph based semi-supervised learning at very low label rates. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1306–1316. PMLR, 2020.

Emmanuel J. Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 2020.

Wei Dai and Olgica Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230–2249, May 2009.

Jiayi Deng, Xiaodong Yang, Jun Yu, Jun Liu, Zhaiming Shen, Danyang Huang, and Huimin Cheng. Network tight community detection. In *Forty-first International Conference on Machine Learning*, 2024.

David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

A. El Alaoui, X. Cheng, A. Ramdas, M. J. Wainwright, and M. I. Jordan. Asymptotic behavior of l_p-based laplacian regularization in semi-supervised learning. In *Conference on Learning Theory (COLT)*, pages 879–906, 2016.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226–231, 1996.

Renzhong Feng, Aitong Huang, Ming-Jun Lai, and Zhaiming Shen. Reconstruction of sparse polynomials via quasi-orthogonal matching pursuit method. *Journal of Computational Mathematics*, 2021.

Mauricio Flores, Jeff Calder, and Gilad Lerman. Analysis and algorithms for ℓ_p-based semi-supervised learning on graphs. *Applied and Computational Harmonic Analysis*, 60:77–122, 2022.

Kimon Fountoulakis, Di Wang, and Shenghao Yang. p-norm flow diffusion for local graph clustering. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3222–3232, 2020.

Matthew A. Herman and Thomas Strohmer. General deviants: An analysis of perturbations in compressed sensing. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):342–349, 2010.

Chester Holtz, Pengwen Chen, Zhengchao Wan, Chung-Kuan Cheng, and Gal Mishne. Continuous partitioning for graph-based semi-supervised learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. NeurIPS 2024.

M. Jacobs, E. Merkurjev, and S. Esedoglu. Auction dynamics: A volume constrained mbo scheme. *Journal of Computational Physics*, 354:288–310, 2018.

Sung Ha Kang, Berta Sandberg, and Andy M Yip. A regularized k-means and multiphase scale segmentation. *Inverse Problems and Imaging*, 5(2):407–429, 2011.

Sung Ha Kang, Behrang Shafei, and Gabriele Steidl. Supervised and transductive multi-class segmentation using p-laplacians and rkhs methods. *Journal of Visual Communication and Image Representation*, 25(5):1136–1148, 2014.

Kyle Kloster and David F. Gleich. Heat kernel based community detection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1386–1395. ACM, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images, 2009.

Ming-Jun Lai and Daniel Mckenzie. Compressive sensing for cut improvement and local clustering. *SIAM Journal on Mathematics of Data Science*, 2(2):368–395, 2020.

Ming-Jun Lai and Zhaiming Shen. A quasi-orthogonal matching pursuit algorithm for compressive sensing. *arXiv preprint arXiv:2007.09534*, 2020.

Ming-Jun Lai and Zhaiming Shen. A compressed sensing based least squares approach to semi-supervised local cluster extraction. *Journal of Scientific Computing*, 94(3):63, 2023.

Kevin Lang and Satish Rao. A flow-based method for improving the expansion or conductance of graph cuts. In *Proceedings of the International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 325–337, 2004.

Haifeng Li. Improved analysis of sp and cosamp under total perturbations. *EURASIP Journal on Advances in Signal Processing*, 2016:1–6, 2016.

J. MacQueen. Classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 14, 2001.

Frank Nielsen. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*, pages 195–211. 2016.

Yaghoub Rahimi, Sung Ha Kang, and Yifei Lou. A lifted ℓ_1 framework for sparse recovery. *Information and Inference: A Journal of the IMA*, 13(1):iaad055, 2024.

Zhaiming Shen, Ming-Jun Lai, and Sheng Li. Graph-based semi-supervised local clustering with few labeled nodes. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4190–4198, 2023.

Pan Shi, Kun He, David Bindel, and John E. Hopcroft. Locally-biased spectral approximation for community detection. *Knowledge-Based Systems*, 164:459–472, 2019.

Z. Shi, S. Osher, and W. Zhu. Weighted nonlocal laplacian on interpolation from sparse data. *Journal of Scientific Computing*, 73(2-3):1164–1177, 2017.

Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Computing*, 42(1):1–26, 2013. doi: 10.1137/110853996.

Joel A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50:2231–2242, 2004.

George C Tseng and Wing H Wong. Tight clustering: a resampling-based approach for identifying stable and tight patterns in data. *Biometrics*, 61(1):10–16, 2005.

Nate Veldt, Christine Klymko, and David F. Gleich. Flow-based local graph clustering with better seed set inclusion. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 378–386. SIAM, 2019.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

James D. Wilson, Simi Wang, Peter J. Mucha, Shankar Bhamidi, and Andrew B. Nobel. A testing based extraction algorithm for identifying significant communities in networks. *The Annals of Applied Statistics*, 8:1853–1891, 2014.

Han Xiao, Karim Rasul, and Roland Vollgraf. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *ICML'16*, pages 40–48. PMLR, 2016. Also introduces the Planetoid benchmark with standardized splits for Cora, CiteSeer, and PubMed datasets.

Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 17, 2004.

D. Zhou and B. Scholkopf. Learning from labeled and unlabeled data using random walks. In *Joint Pattern Recognition Symposium*, pages 237–244. Springer, 2004.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328. MIT Press, 2004a.

D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Scholkopf. Ranking on data manifolds. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 169–176. MIT Press, 2004b.

D. Zhou, J. Huang, and B. Scholkopf. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 1036–1043. ACM, 2005.

X. Zhou and M. Belkin. Semi-supervised learning by higher order regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 892–900, 2011.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation, 2002. ProQuest number: information to all users.

Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 912–919, 2003.

A Lemmas and deferred proofs

Let us introduce

$$\mathbf{x}^* := \underset{\mathbf{x}\in\mathbb{R}^{|V|-|U|}}{\arg\min} \{\|L^{in}_{V\setminus U}\mathbf{x} - \mathbf{y}^{in}\|_2 : \|\mathbf{x}\|_0 \le \hat{n}_1 - |U|\}, \tag{1}$$

where $\mathbf{y}^{in} := -\sum_{i\in V\setminus U} \ell_i^{in} = L^{in}\mathbf{1}_{V\setminus U}$, and

$$\mathbf{x}^{\#} := \underset{\mathbf{x}\in\mathbb{R}^{|V|-|U|}}{\arg\min} \{\|L_{V\setminus U}\mathbf{x} - \mathbf{y}\|_2 : \|\mathbf{x}\|_0 \le \hat{n}_1 - |U|\}, \tag{2}$$

where $\mathbf{y} := -\sum_{i\in V\setminus U} \ell_i = L\mathbf{1}_{V\setminus U}$. The solution $\mathbf{x}^*$ gives the underlying true node indices associated with L^{in}, while $\mathbf{x}^{\#}$ gives the algorithmic output node indices associated with L.

We highlight the following Lemma 1, which is a crucial step in establishing the convergence result in LCE. We relax the assumption from $\|\Delta L\|_2 = o(n^{-1/2})$ (see Lemma 4 in Appendix C) [Shen et al., 2023] to $\|\Delta L\|_2 = o(1)$.

Lemma 1 *Let $\Delta L := L - L^{in}$. Suppose $U \subset C_1$ and $0.1|C_1| < |U| < 0.9|C_1|$. Suppose further that $\|\Delta L\|_2 = o(1)$ and $\left(\delta_{3(|C_1|-|U|)}(L)\right)^{\log(n)} = o(1)$. Then*

$$\frac{\|\mathbf{x}^{\#} - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2} = o(1). \tag{3}$$

Lemma 1 establishes the convergence result between $\mathbf{x}^{\#}$ and $\mathbf{x}^*$ when ΔL is small, which is the key step for establishing the correctness of our algorithms in Theorem 1.

Lemma 2 *Suppose G satisfies Assumptions 1 - 3. Let v be any sampled node with $v \in C_s$ for some $s \ge 1$. Then, as n gets large, the cardinality of the set $S := \{i \in C_r : \mathrm{comembership}(\mathbf{1}_{C^{\#}})_{v,i} = \mathrm{comembership}(\mathbf{1}_{C_s})_{v,i}\}$ satisfies*

$$\begin{cases} |S| \ge (1 - o(1))n_{\min}, & \text{if } r = s, \\ |S| \le o(n_{\min}), & \text{otherwise.} \end{cases}$$

A.1 Proof of Lemma 1

Proof. [Proof of Lemma 1] First, since $\|L - L^{in}\|_2 = o(1)$, we have

$$\frac{\|\Delta\mathbf{y}\|_2}{\|\mathbf{y}\|_2} = \frac{\|\mathbf{y} - \mathbf{y}^{in}\|_2}{\|\mathbf{y}\|_2} = \frac{\|(L - L^{in})\mathbf{1}_{V\setminus U}\|_2}{\|L\mathbf{1}_{V\setminus U}\|_2} \le \frac{o(1)\sqrt{n}}{\|L\mathbf{1}_U\|_2}.$$

By Assumption 2,, the cluster C_1 is on the same order as the size of the graph n, hence $U \subset C_1$ is also on the same order as n. By Assumption 3, all the nodes asymptotically have the same degree, therefore $\|L\mathbf{1}_U\|_2 = \Theta(\|\mathbf{1}_U\|_2) = \Theta(\sqrt{n})$. Hence

$$\frac{\|\mathbf{y} - \mathbf{y}^{in}\|_2}{\|\mathbf{y}\|_2} \le \frac{o(1)\sqrt{n}}{\Theta(\sqrt{n})} = o(1).$$

Therefore, the quantity $\epsilon_{\mathbf{y}} = o(1)$ in Lemma 5.

With the same assumption $\|L - L^{in}\|_2 = o(1)$, it is not hard to see that, if the singular values of L decays at a reasonable rate, then by applying the eigenvalue interlacing theorem, we have $\epsilon^s_\Phi = o(1)$ as well (by letting $\Phi = L$ in Lemma 5 in Appendix D). With these, the second term on the right-hand-side of the inequality in Lemma 5 satisfies

$$\tau\frac{\sqrt{1+\delta_s}}{1-\epsilon^s_\Phi}(\epsilon^s_\Phi + \epsilon_{\mathbf{y}}) \le o(1).$$

Furthermore, since $\mathbf{x}^{\#}$ is the output of Algorithm 4 after $m = O(\log n)$ iteration, we have $\rho^m = O\left((\delta_{3(|C_1|-|U|)}(L))^{\log n}\right) = o(1)$. Putting these together gives

$$\frac{\|\mathbf{x}^{\#} - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2} = o(1)$$

as desired. □

A.2 Proof of Theorem 1

Proof. [Proof of Theorem 1] For any $s \geq 1$, let us case on whether $v \in C_s$ or not. If $v \in C_s$, then by Lemma 1 and Theorem 4, we have

$$|C^{\#} \triangle (\tilde{C}_s \cap C^{\#})| \leq |C^{\#} \triangle (C_s \cap C^{\#})| + |(C_s \triangle \tilde{C}_s) \cap C^{\#}| \leq o(|C^{\#}|) + o(|C^{\#}|) = o(|C^{\#}|).$$

Therefore, $|\tilde{C}_s \cap C^{\#}| \geq |C^{\#}| - o(|C^{\#}|) = (1 - o(1))|C^{\#}|$.

If $v \in C_t, t \neq s$, then $|C_s \cap C^{\#}| \leq o(|C^{\#}|)$. We have

$$|C^{\#} \triangle (\tilde{C}_s \cap C^{\#})| \geq |C^{\#} \triangle (C_s \cap C^{\#})| - |(C_s \triangle \tilde{C}_s) \cap C^{\#}| \geq |C^{\#}| - o(|C^{\#}|) - o(|C^{\#}|) = |C^{\#}| - o(|C^{\#}|).$$

Therefore, $|\tilde{C}_s \cap C^{\#}| \leq |C^{\#}| - (|C^{\#}| - o(|C^{\#}|)) = o(|C^{\#}|)$.

□

A.3 Proof of Lemma 2

Proof. [Proof of Lemma 2] By Lemma 1 and Theorem 4, we have $|C^{\#} \triangle (C_s \cap C^{\#})| \leq o(C^{\#}) = o(n_{\min})$. Therefore,

$$|C^{\#} \triangle C_s| = |C^{\#} \triangle (C_s \cap C^{\#})| + |C_s \setminus C^{\#}| \leq o(n_{\min}) + n_s - n_{\min}.$$

Hence

$$\begin{aligned} |\{i \in C_s : \text{comembership}(\mathbf{1}_{C^{\#}})_{v,i} \neq \text{comembership}(\mathbf{1}_{C_s})_{v,i}\}| &= |(C^{\#} \triangle C_s) \cap C_s| \leq |C^{\#} \triangle C_s| \\ &\leq o(n_{\min}) + n_s - n_{\min}. \end{aligned}$$

So we conclude

$$\begin{aligned} |\{i \in C_s : \text{comembership}(\mathbf{1}_{C^{\#}})_{v,i} = \text{comembership}(\mathbf{1}_{C_s})_{v,i}\}| &\geq n_s - (o(n_{\min}) + n_s - n_{\min}) \\ &= n_{\min} - o(n_{\min}). \end{aligned}$$

As $C_s \cap C_t = \emptyset$, we have $|C^{\#} \cap C_t| \leq o(n_{min})$. Therefore,

$$|\{i \in C_t : \text{comembership}(\mathbf{1}_{C^{\#}})_{v,i} = \text{comembership}(\mathbf{1}_{C_t})_{v,i}\}| \leq o(n_{\min}).$$

□

A.4 Proof of Proposition 1

Proof. [Proof of Proposition 1] By Lemma 2, we have the following estimates for the entries in the comembership matrix M asymptotically.

For any $i = j \in C_s$, some $s \geq 1$, the entries in M satisfies

$$M_{i,i} \geq \frac{n_s}{n} \cdot \frac{(1 - o(1))n_{\min}}{n_s} = (1 - o(1))\frac{n_{\min}}{n}.$$

For any distinct $i, j \in C_s$, $s \geq 1$, by Assumption 3, the entries in M satisfies

$$M_{i,j} \geq \frac{1}{n} \cdot \frac{n_{\min}^2(1 - o(1))}{n} = (1 - o(1))\frac{n_{\min}^2}{n^2}.$$

For any distinct i, j such that $i \in C_s$, $j \in C_t$, $s \neq t$, the entries in M satisfies

$$\begin{aligned} M_{i,j} &\leq \frac{n_s}{n} \cdot \frac{(1 - o(1))n_{\min}}{n_s} \cdot \frac{o(n_{\min})}{n - n_s} + \frac{n_t}{n - n_t} \cdot \frac{(1 - o(1))n_{\min}}{n_t} \cdot \frac{o(n_{\min})}{n} \\ &\leq \frac{2n_{\min}^2(1 - o(1)) \cdot o(1)}{n^2} + \frac{2n_{\min}^2(1 - o(1)) \cdot o(1)}{n^2} \\ &= o\left(\frac{4n_{\min}^2(1 - o(1))}{n^2}\right). \end{aligned}$$

□

A.5 Proof of Theorem 2

Proof. [Proof of Theorem 2] By direct computation, we can choose $\delta = \frac{1}{2}(\frac{n_{min}}{n})^2$ such that

$$(1 - o(1))\frac{n_{\min}}{n} > \delta > o\left(\frac{4n_{\min}^2(1 - o(1))}{n^2}\right).$$

and

$$(1 - o(1))\frac{n_{\min}^2}{n^2} > \delta > o\left(\frac{4n_{\min}^2(1 - o(1))}{n^2}\right).$$

Hence, for any $i \in C_s$ satisfies $M_{v,i} > \delta$, and any $i \in C_t, t \neq s$, satisfies $M_{v,i} < \delta$. So we conclude $C_s^{\#} = C_s$. □

Remark 1 *Note that All the statements in Theorem 1 and 2, Lemma 1 and 2, and Proposition 1, are asymptotic statements. The $o(\cdot)$ notation is with respect to $n \to \infty$.*

B Additional Algorithms

Algorithm 3 Semi-supervised Local Clustering (SSLC) for Multiple Clusters

Require: The adjacency matrix A of an underlying graph G, the number of cluster k, the initial seed(s) Γ_s for each cluster, the estimated size n_s for each cluster, $s = 1, \cdots, k$, the number of resampling iteration ℓ

Ensure: desired output clusters $C_1^{\#}, \cdots, C_k^{\#}$

1: **for** $s = 1 : k$ **do**
2: $\quad \tilde{C}_s \leftarrow \mathrm{LCE}(A, n_s, \Gamma_s)$
3: **end for**
4: **for** $i = 1 : \ell$ **do**
5: $\quad v \leftarrow$ uniform randomly sampled seed node from G
6: $\quad C^{\#} \leftarrow \mathrm{LCE}(A, \min\{n_s\}_{s \geq 1}, v)$
7: $\quad$ **if** $|\tilde{C}_s \cap C^{\#}| > 0.5|C^{\#}|$ for some s **then**
8: $\quad\quad \Gamma_s \leftarrow \Gamma_s \cup \{v\}$
9: $\quad\quad \tilde{C}_s \leftarrow \mathrm{LCE}(A, n_s, \Gamma_s)$
10: $\quad$ **end if**
11: **end for**
12: **for** $s = 1 : k$ **do**
13: $\quad C_s^{\#} \leftarrow \tilde{C}_s$
14: **end for**

We use $C^{\#}$ to indicate the output of one of the LCE steps after sampling a random node v. Since v can be from any cluster s, the notation $C^{\#}$ is independent of s. Due to Theorem 1, the condition $|\tilde{C}_s \cap C^{\#}| > 0.5|C^{\#}|$ can only happen for one particular s. Therefore, Step 7 in Algorithm 3 will only add each v to one of the clusters.

Remark 2 *One key advantage of Algorithm 3 is its ability to simultaneously identify all clusters, while previous method such as LCE can only detect one cluster at a time. This characteristic provides greater flexibility and makes Algorithm 3 significantly more efficient for practical applications, particularly when the number of clusters is large. This advantage is also reflected in the first and second columns of Figure 3.*

C Review on Local Cluster Extraction (LCE)

The following result is central to our proposed sparse solution based local clustering method. We omit its proof by referring to Von Luxburg [2007].

Lemma 3 *Let G be an undirected graph with non-negative weights. The multiplicity k of the eigenvalue zero of the graph Laplacian $L := I - D^{-1}A$ equals to the number of connected components $C_1, C_2, \cdots, C_k$ in G. Further, the indicator vectors $\mathbf{1}_{C_1}, \cdots, \mathbf{1}_{C_k} \in \mathbb{R}^n$ on these components span the kernel of L.*

For the convenience of our discussion, let us introduce more notations . For a graph $G = (V, E)$ with certain underlying community structure, it is convenient to write $G = G^{in} \cup G^{out}$, where $G^{in} = (V, E^{in})$, $G^{out} = (V, E^{out})$. Here E^{in} is the set of all intra-connection edges within the same community (cluster), E^{out} is the remaining edges in E. Further, we use A^{in} and L^{in} to denote the adjacency matrix and Laplacian matrix associated with G^{in} respectively. In practice, we do not guarantee knowledge of the cluster to which each individual vertex belongs, meaning that A^{in} and L^{in} are not directly accessible. Instead, we only have access to A and L. A summary of the notations being used throughout this paper is included in Table 7 in Appendix G.

Let us first briefly introduce the idea Local Cluster Extraction (LCE), which applies the idea of compressive sensing (or sparse solution) technique to extract the target cluster in a semi-supervised manner. See also [Lai and Mckenzie, 2020, Lai and Shen, 2023, Shen et al., 2023] for references.

Suppose the graph G consists of k connected equal-size components $C_1, \cdots, C_k$, in other words, there is no edge connection between different clusters, i.e., $L = L^{in}$. For illustration purpose, let us permute the matrix according to the membership of each node, then L^{in} is in a block diagonal form i.e., all the off-diagnoal blocks equal to zero:

$$L = L^{in} = \begin{pmatrix} L^{in}_{C_1} & & & \\ & L^{in}_{C_2} & & \\ & & \ddots & \\ & & & L^{in}_{C_k} \end{pmatrix}. \tag{4}$$

Suppose that the target cluster is C_1. By Lemma 3, $\{\mathbf{1}_{C_1}, \cdots, \mathbf{1}_{C_k}\}$ forms a basis of the kernel W_0 of L. Note that all the $\mathbf{1}_{C_i}$ have disjoint supports, so for $\mathbf{w} \in W_0$ and $\mathbf{w} \neq 0$, we have write

$$\mathbf{w} = \sum_{i=1}^{k} \alpha_i \mathbf{1}_{C_i} \tag{5}$$

with some $\alpha_i \neq 0$. If a prior knowledge is given that first node $v_1 \in C_1$, then the first cluster C_1 can be found by solving

$$\min ||\mathbf{w}||_0 \quad \text{s.t.} \quad L^{in}\mathbf{w} = \mathbf{0} \quad \text{and} \quad w_1 = 1, \tag{6}$$

which gives the solution $\mathbf{w} = \mathbf{1}_{C_1} \in \mathbb{R}^n$ as desired. It is worthwhile to note that (6) is equivalent to

$$\min ||\mathbf{w}_{-1}||_0 \quad \text{s.t.} \quad L^{in}_{-1}\mathbf{w}_{-1} = -\ell_1, \tag{7}$$

where L^{in}_{-1} is a submatrix of L^{in} with first column being removed, ℓ_1 is the first column of L^{in}. The solution to (7) is $\mathbf{w}_{-1} = \mathbf{1}_{C_1 \setminus v_1} \in \mathbb{R}^{n-1}$ which encodes the same index information as the solution to (6). The benefit of formulation (7) is that it can be solved by compressive sensing algorithms. Note that the sparse solution from solving (6) or (7) always gives the indices corresponding to the target cluster. Therefore, the permutation does not affect our analysis and clustering result. The above procedure is named CS-LCE or LCE in [Shen et al., 2023]. We summarize LCE as Algorithm 4 in Appendix C and provide brief analysis. We also briefly introduce the compressive sensing and sparse solution technique in Appendix D.

The following results are established in [Shen et al., 2023], and we refer the proofs to [Shen et al., 2023]. Let L^{in} be the matrix as in (4), then the true solution $\mathbf{x}^*$ which encodes the local cluster information is

$$\mathbf{x}^* := \underset{\mathbf{x} \in \mathbb{R}^{|V|-|U|}}{\arg\min} \{\|L^{in}_{V \setminus U}\mathbf{x} - \mathbf{y}^{in}\|_2 : \|\mathbf{x}\|_0 \leq \hat{n}_1 - |U|\} \tag{8}$$

where $\mathbf{y}^{in} = -\sum_{i \in V \setminus U} \ell^{in}_i = L^{in}\mathbf{1}_{V \setminus U}$.

Theorem 3 *Suppose $U \subset C_1$ and $0.1|C_1| < |U| < 0.9|C_1|$. Then $\mathbf{x}^* = \mathbf{1}_{C_1 \setminus U} \in \mathbb{R}^{|V|-|U|}$ is the unique solution to (8).*

Lemma 4 *Let $\Delta L := L - L^{in}$. Suppose $U \subset C_1$ and $0.1|C_1| < |U| < 0.9|C_1|$. Suppose further that $\|\Delta L\|_2 = o(n^{-\frac{1}{2}})$ and $\delta_{3(|C_1|-|U|)}(L) = o(1)$. Then*

$$\frac{\|\mathbf{x}^{\#} - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2} = o(1). \tag{9}$$

Theorem 4 *Under the same assumption as Lemma 4. Then*

$$\frac{|C_1 \triangle C_1^{\#}|}{|C_1|} \leq o(1). \tag{10}$$

We make use of Lemma 1 (improved version of Lemma 4) and Theorem 4 in our main proofs. The LCE algorithm is summarized in Algorithm 4.

Algorithm 4 Compressive Sensing of Local Cluster Extraction (CS-LCE or LCE [Shen et al., 2023])

Require: Adjacency matrix A, the seed(s) set $\Gamma \subset C$ for some target cluster C, a known estimated size $\hat{n}_1 \approx |C_1|$, a fixed number of random walk depth $t \in \mathbb{Z}^+$, sparsity parameter $\gamma \in [0.1, 0.5]$, random walk threshold parameter $\epsilon \in (0, 1)$, rejection parameter $R \in [0.1, 0.9]$.

Ensure: the output target cluster $C^{\#}$

1: Compute $L = I - D^{-1}A$, $P = AD^{-1}$, $\mathbf{v}^0 = D\mathbf{1}_\Gamma$ and $\mathbf{v}^{(t)} = P^t\mathbf{v}^{(0)}$

2: Define $\Omega = \mathcal{L}_{(1+\epsilon)\hat{n}}(\mathbf{v}^{(t)})$

3: Let U be the set of column indices of $\gamma \cdot |\Omega|$ smallest components of the vector $|L_\Omega^\top| \cdot |L\mathbf{1}_\Omega|$.

4: Set $\mathbf{y} := -\sum_{i \in V\setminus U} \ell_i = L\mathbf{1}_{V\setminus U}$. Let $\mathbf{x}^{\#}$ be the solution to

$$\underset{\mathbf{x}\in\mathbb{R}^{|V|-|U|}}{\arg\min} \{\|L_{V\setminus U}\mathbf{x} - \mathbf{y}\|_2 : \|\mathbf{x}\|_0 \leq \hat{n}_1 - |U|\} \tag{11}$$

obtained by using $O(\log n)$ iterations of *Subspace Pursuit* Dai and Milenkovic [2009].

5: Let $C^{\#} = \{i : \mathbf{x}_i^{\#} > R\} \cup U$

The notation $\mathcal{L}$ is defined as

$$\mathcal{L}_s(\mathbf{v}) := \{i \in [n] : v_i \text{ among } s \text{ largest-in-magnitude entries in } \mathbf{v}\}.$$

D Background on Compressive Sensing and Sparse Solution Technique

The concept of compressive sensing (also called compressed sensing or sparse sampling) emerged from fundamental challenges in signal acquisition and efficient data compression. At its core, it addresses the inverse problem of recovering a sparse (or compressible) signal from a small number of noisy linear measurements:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{x}\|_0 \quad s.t. \quad \|\Phi\mathbf{x} - \mathbf{y}\|_2 \leq \epsilon, \tag{12}$$

where $\Phi \in \mathbb{R}^{m\times n}$ is called sensing matrix (usually underdetermined), $\mathbf{y} \in \mathbb{R}^n$ is called measurement vector, and the "zero quasi-norm" $\|\cdot\|_0$ counts the number of nonzero components in a vector. Among the key contributors, Donoho [Donoho, 2006] and Candès, Romberg and Tao [Candès et al., 2006] are widely credited with being the first to explicitly introduce this concept and make it popular. Since then, two families of approaches such as thresholding type of algorithms [Blumensath and Davies, 2009] and greedy type of algorithms [Tropp, 2004, Feng et al., 2021, Lai and Shen, 2020] have been developed based on the idea of compressive sensing. One particular type of greedy algorithms that has garnered our attention is the subspace pursuit [Dai and Milenkovic, 2009].

One of the reasons behind the remarkable usefulness of compressive sensing lies in its robustness against errors, including both additive and multiplicative types. More precisely, suppose we know $\mathbf{y} = \Phi\mathbf{x}^*$ where $\mathbf{y}$ is the exact measurement of the acquired signal and Φ is the exact measurement of the sensing matrix. However, we may only be able to access to the noisy version $\tilde{\mathbf{y}} = \mathbf{y} + \Delta\mathbf{y}$ and $\tilde{\Phi} = \Phi + \Delta\Phi$. In such case, we can still approximate the solution $\mathbf{x}^*$ well from the noisy measurements $\tilde{\mathbf{y}}$ and $\tilde{\Phi}$, as explained in the work of [Herman and Strohmer, 2010]. A unified framework of lifted ℓ_1 form is explored in [Rahimi et al., 2024].One crucial concept which is often employed in the compressive sensing algorithm is called Restricted Isometry Property (RIP).

Definition 1 (Restricted Isometry Property) *Let $0 < s < m$ be an integer, and sensing matrix $\Phi \in \mathbb{R}^{m\times n}$. Suppose there exists a constant $\delta_s > 0$ such that*

$$(1 - \delta_s)\|\mathbf{x}\|_2^2 \leq \|\Phi\mathbf{x}\|_2^2 \leq (1 + \delta_s)\|\mathbf{x}\|_2^2 \tag{13}$$

for all $\mathbf{x} \in \mathbb{R}^n$ with $\|\mathbf{x}\|_0 \leq s$. Then the matrix Φ is said to have the Restricted Isometry Property (RIP) of order s. The smallest constant $\delta_s(\Phi)$ which makes (13) hold is called the Restricted Isometry Constant (RIC) of Φ.

For Subspace Pursuit algorithm, we have the result in Lemma 5 [Li, 2016].

Lemma 5 *Let $\mathbf{x}^*$, $\mathbf{y}$, $\tilde{y}$, Φ, $\tilde{\Phi}$ be as defined above, and for any $t \in [n]$, let $\delta_s := \delta_s(\tilde{\Phi})$. Suppose that $\|\mathbf{x}^*\|_0 \leq s$. Define the following constants:*

$$\epsilon_{\mathbf{y}} := \|\mathbf{\Delta y}\|_2 / \|\mathbf{y}\|_2 \quad \textit{and} \quad \epsilon_{\Phi}^s := \|\Delta\Phi\|_2^{(s)} / \|\Phi\|_2^{(s)}$$

where $\|M\|_2^{(s)} := \max\{\|M_S\|_2 : S \subset [n], \#(S) = s\}$ for any matrix M. Define further:

$$\rho := \frac{\sqrt{2\delta_{3s}^2(1+\delta_{3s}^2)}}{1-\delta_{3s}^2} \quad \textit{and} \quad \tau := \frac{(\sqrt{2}+2)\delta_{3s}}{\sqrt{1-\delta_{3s}^2}}(1-\delta_{3s})(1-\rho) + \frac{2\sqrt{2}+1}{(1-\delta_{3s})(1-\rho)}.$$

Assume that $\delta_{3s} < 0.4859$ and let $\mathbf{x}^{(m)}$ be the output of Algorithm 4 after m iterations. Then:

$$\frac{\|\mathbf{x}^* - \mathbf{x}^{(m)}\|_2}{\|\mathbf{x}^*\|_2} \leq \rho^m + \tau \frac{\sqrt{1+\delta_s}}{1-\epsilon_{\Phi}^s}(\epsilon_{\Phi}^s + \epsilon_{\mathbf{y}}).$$

E Implementation Details

E.1 Constructing KNN Graphs

Let $\mathbf{x}_i \in \mathbb{R}^n$ be the vectorization of an image or the feature extracted from an image, we define the following affinity matrix of the K-NN auxiliary graph based on Gaussian kernel:

$$A_{ij} = \begin{cases} e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma_i\sigma_j} & \text{if} \quad \mathbf{x}_j \in NN(\mathbf{x}_i, K), \\ 0 & \text{otherwise.} \end{cases}$$

Note that similar construction has also appeared among others [Zelnik-Manor and Perona, 2004, Jacobs et al., 2018, Calder et al., 2020]. To construct high-quality graphs, we trained autoencoders to extract key features from the image data, we adopt the same parameters as [Calder et al., 2020] for training autoencoders to obtain these features.

The notation $NN(\mathbf{x}_i, K)$ indicates the set of K-nearest neighbours of $\mathbf{x}_i$, and $\sigma_i := \|\mathbf{x}_i - \mathbf{x}_i^{(r)}\|$ where $\mathbf{x}_i^{(r)}$ is the r-th closest point of $\mathbf{x}_i$. Note that the above A_{ij} is not necessary symmetric, so we consider $\tilde{A}_{ij} = A^T A$ for symmetrization. Alternatively, one may also want to consider $\tilde{A} = \max\{A_{ij}, A_{ji}\}$ or $\tilde{A} = (A_{ij} + A_{ji})/2$. We use $\tilde{A}$ as the input adjacency matrix for our algorithms. In our experiments, the local scaling parameters are chosen to be $K = 15$, $r = 10$ for all three real image datasets.

E.2 Parameters of Synthetic Data Generating and Algorithms

In all implementations where the LCE are applied, we sampled the seeds Γ_i uniformly from each C_i for all the implementations. We fix the rejection parameter $R = 0.1$, the random walk depth $t = 3$, random walk threshold parameter $\epsilon = 0.8$, and the removal set parameter $\gamma = 0.2$ for all experiments.

For synthetic data, the symmetric stochastic block model consists of three equal size clusters of $n_1 = 200, 400, 600, 800, 1000$ with connection probability $p = 5\log(3n_1)/(3n_1)$ and $q = \log(3n_1)/(3n_1)$. The general stochastic block model consists of clusters' sizes as $\mathbf{n} = (n_1, 2n_1, 5n_1)$ where n_1 is chosen from $200, 400, 600, 800, 1000$, and the connection probability has the matrix form

$$P = [P_{11}, P_{12}, P_{13}; P_{21}, P_{22}, P_{23}; P_{31}, P_{32}, P_{33}]$$

with $P_{11} = \log^2(8n_1)/(6n_1)$, $P_{22} = \log^2(8n_1)/(12n_1)$, $P_{33} = \log^2(8n_1)/(30n_1)$, $P_{12} = P_{21} = P_{13} = P_{31} = P_{23} = P_{32} = \log(8n_1)/(6n_1)$. In the implementation of SSLC on synthetic data, we choose the iteration number $L = 60$ in the symmetric case and $L = 90$ in the nonsymmetric case.

In the implementation of SSLC on FashionMNIST and CIFAR-10 (both with and w/o outliers), we choose the iteration number to be $L = 50k$ where k is the number of seeds, i.e., $k = 1, 2, 3, 4, 5$. In the implementation of USLC on FashionMNIST and CIFAR-10 (both with and w/o outliers), we choose the iteration number to be $L = 1000$, and we set $\delta = 0.01$ for FashionMNIST and $\delta = 0.005$ for CIFAR-10.

E.3 Geometric Dataset

Three Lines We generate three parallel lines in the x-y plane defined by:

- Line 1: $y = 0$ with $x \in [0, 6]$
- Line 2: $y = 1$ with $x \in [0, 6]$
- Line 3: $y = 2$ with $x \in [0, 6]$

For each line, we sample **1,200 points** uniformly at random. The embedding into $\mathbb{R}^{100}$ is performed as:

1. **Zero-padding**: $(x, y) \mapsto (x, y, 0, \ldots, 0) \in \mathbb{R}^{100}$
2. **Noise addition**: Each coordinate z_i is perturbed as $z_i \leftarrow z_i + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, 0.15)$

Three Circles We construct three concentric circles with:

- Circle 1: Radius $r = 1.0$ (500 points)
- Circle 2: Radius $r = 2.4$ (1,200 points)
- Circle 3: Radius $r = 3.8$ (1,900 points)

Points are sampled uniformly along each circle, totaling **3,600 points**. The $\mathbb{R}^{100}$ embedding follows the same zero-padding and noise injection procedure as above.

Three Moons Three semicircular clusters are generated with:

- Moon 1: Upper semicircle, radius 1.0, centered at $(0, 0)$ (1,200 points)
- Moon 2: Lower semicircle, radius 1.5, centered at $(1.5, 0.4)$ (1,200 points)
- Moon 3: Upper semicircle, radius 1.0, centered at $(3, 0)$ (1,200 points)

Each dataset uses identical embedding:

$$(x, y) \mapsto (x, y, 0, \ldots, 0) + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.15^2 \mathbf{I}_{100})$$

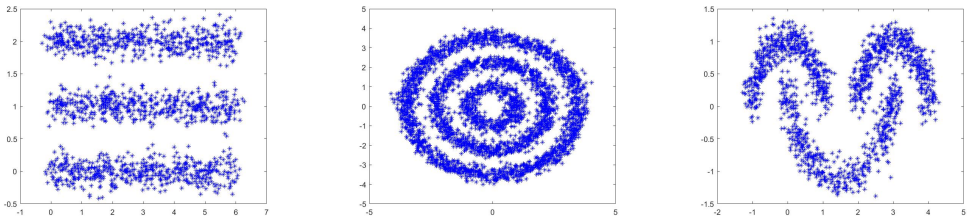


Figure 4: 2D Visulization of Geometric Dataset

Table 5: Jaccard Index of SSLC compared against the spectral norm of ΔL and SNR value on SSBM(n, k, p, q) with $k = 3$, $p = 6 \log n/n$, $q = \log n/n$.

n	Jaccard Index (%)	Spectral norm of ΔL	SNR
100	84.59	0.4543	4.80
200	93.73	0.4238	5.52
400	96.04	0.3982	6.24
800	99.18	0.3725	6.96

F Additional Experimental Results

Table 6: Average accuracy and standard deviation on the largest connected subgraph over 100 trials

	# Labels per class	0	1	3	5	10
Cora	Laplace (LP)	-	21.8 (14.3)	37.6 (12.3)	51.3 (11.9)	66.9 (6.8)
	Poisson	-	59.8 (7.9)	66.2 (5.8)	72.4 (2.1)	74.1 (1.8)
	PoissonMBO	-	59.9 (6.4)	69.1 (3.1)	72.4 (2.4)	74.3 (2.1)
	CutSSL	-	67.4 (3.4)	73.2 (3.1)	75.8 (2.1)	**78.7 (1.1)**
	SSLC/USLC	**64.2 (5.7)**	**69.2 (4.0)**	**75.5 (3.4)**	**76.9 (2.6)**	77.9 (1.3)
CiteSeer	Laplace (LP)	-	27.9 (10.4)	47.6 (8.1)	56.0 (5.9)	63.7 (3.5)
	Poisson	-	59.4 (5.4)	59.4 (5.4)	62.7 (4.2)	66.9 (1.8)
	PoissonMBO	-	47.7 (8.0)	55.7 (3.2)	61.0 (1.7)	63.1 (1.7)
	CutSSL	-	62.4 (4.6)	63.4 (7.2)	66.9 (1.4)	68.1 (1.3)
	SSLC/USLC	**59.8 (5.7)**	**65.2 (5.3)**	**67.1 (4.7)**	**68.6 (2.9)**	**69.3 (1.6)**
PubMed	Laplace (LP)	-	34.6 (8.8)	35.7 (8.2)	36.9 (8.1)	39.6 (9.1)
	Poisson	-	56.7 (12.8)	66.5 (6.6)	68.4 (5.9)	71.2 (3.4)
	PoissonMBO	-	56.9 (7.3)	67.9 (3.4)	69.6 (3.1)	71.4 (2.5)
	CutSSL	-	63.1 (4.7)	70.4 (3.1)	72.8 (2.9)	**74.1 (1.4)**
	SSLC/USLC	**61.6 (6.2)**	**67.3 (5.1)**	**71.7 (3.9)**	**73.1 (2.9)**	73.9 (2.2)

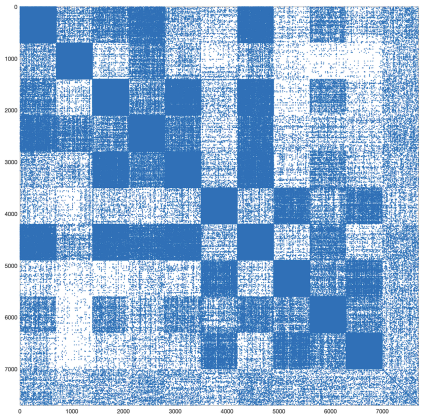


Figure 5: Affinity matrix of FashionMNIST after adding the outlier images (the last block consists of 10% outliers compared to the size of original dataset)

G Notations

Table 7: Table of Notations

Symbols	Interpretations
G	general graph of interest
E	set of edges of graph G
V	set of nodes in G (size denoted by n)
C_s	each underlying true cluster
$C_s^{\#}$	each extracted cluster from algorithm
Γ_s	set of Seeds for each cluster
U	removal set from V in Algorithm 4
G^{in}	subgraph of G on V with edge set E^{in}
G^{out}	subgraph of G on V with edge set E^{out}
E^{in}	subset of E which consists only intra-connection edges
E^{out}	the complement of E^{in} within E
A (A^{in})	adjacency matrix of G (G^{in})
L (L^{in})	random walk Laplacian matrix of G (G^{in})
L_C (L_C^{in})	submatrix of L (L^{in}) with column indices $C \subset V$
ℓ_i (ℓ_i^{in})	i-th column of L (L^{in})
L_Ω^{in}	submatrix of L^{in} with column indices $\Omega \subset V$
$\lvert M \rvert$	entrywised absolute value operation on matrix M
$\lVert M \rVert_2$	$\lVert \cdot \rVert_2$ norm of matrix M
$\lvert \mathbf{v} \rvert$	entrywised absolute value operation on vector $\mathbf{v}$
$\lVert \mathbf{v} \rVert_2$	$\lVert \cdot \rVert_2$ norm of vector $\mathbf{v}$.
$\mathbf{1}_C$	indicator vector on subset $C \subset V$
$\triangle$	set symmetric difference
$\mathcal{L}_s(\mathbf{v})$	$\{i \in [n] : v_i$ among s largest-in-magnitude entries in $\mathbf{v}\}$