Scalable Semi-Supervised Clustering via Structural Entropy with Different Constraints

Guangjie Zeng, Hao Peng, Angsheng Li, Jia Wu, Chunyang Liu, Philip S. Yu, *Fellow, IEEE*

***Abstract*—Semi-supervised clustering leverages prior information in the form of constraints to achieve higher-quality clustering outcomes. However, most existing methods struggle with large-scale datasets owing to their high time and space complexity. Moreover, they encounter the challenge of seamlessly integrating various constraints, thereby limiting their applicability. In this paper, we present Scalable Semi-supervised clustering via Structural Entropy (SSSE), a novel method that tackles scalable datasets with different types of constraints from diverse sources to perform both semi-supervised partitioning and hierarchical clustering, which is fully explainable compared to deep learning-based methods. Specifically, we design objectives based on structural entropy, integrating constraints for semi-supervised partitioning and hierarchical clustering. To achieve scalability on data size, we develop efficient algorithms based on graph sampling to reduce the time and space complexity. To achieve generalization on constraint types, we formulate a uniform view for widely used pairwise and label constraints. Extensive experiments on real-world clustering datasets at different scales demonstrate the superiority of SSSE in clustering accuracy and scalability with different constraints. Additionally, Cell clustering experiments on single-cell RNA-seq datasets demonstrate the functionality of SSSE for biological data analysis.**

***Index Terms*—Semi-supervised clustering, structural entropy, scalable clustering, biological data analysis**

I. Introduction

Semi-supervised clustering extends unsupervised clustering with additional prior information to enhance the clustering outcomes with higher quality [1]. Conventional unsupervised clustering aims to divide data points into groups with high intra-group similarities and low inter-group similarities [2] relying solely on the input data. In contrast, semi-supervised clustering techniques guide the clustering process by harnessing the power of prior information in the form of constraints [3], making clustering outcomes more accurate and better aligned to user preference. Over the past years, semi-supervised clustering has achieved success in many domains, such as image segmentation [4], bioinformatics [5], and medicine [6].

Most semi-supervised clustering methods are derived from conventional unsupervised techniques by integrating prior information, such as KMeans-based [7], hierarchical-based [8], density-based [9], spectral clustering-based [10], nonnegative matrix factorization (NMF)-based [11], and neural network-based [5] methods. These provided prior information can come from various sources and manifest in different types of constraints. Two commonly used constraint types are pairwise constraint [12] and label constraint [1], [7]. Other types of constraints, such as triplet constraint [13] and size constraint [14], also exist. However, many existing methods are tailored to handle a single type of constraint, which hinders their generalization ability. The lack of ability to handle different types of constraints is a major limitation of existing semi-supervised clustering methods.

The key to semi-supervised clustering lies in how they utilize prior information. A prevalent way involves incorporating a regularization term into the original clustering objective [7], [10]. This regularization term prevents algorithms from forming clusters conflicting with the provided prior information. Another way involves a similarity (or distance) learning step using prior information along with a clustering step [15], [16]. Similarity learning gives more informative similarities among instances, guiding the downstream clustering step to a more accurate outcome. Additionally, prior information can be used in forming the initialization of KMeans clustering [1], guiding the merging of agglomerative clustering [17], and so on. However, regardless of how prior information is utilized, most semi-supervised clustering methods exhibit significant time and space complexity. For methods [7], [10] based on regularization terms, they tend to consume more time and space than the original clustering methods. For methods [15], [16] based on similarity learning, they require time-consuming instance-wise learning. Consequently, neither of them can be applied to large-scale datasets. Many efforts have been made to achieve scalable clustering, such as over-clustering data points into super-instances [18] and selecting representative anchors for relationship measuring [19], [20]. In contrast, the scalability issue of semi-supervised clustering methods is less addressed. Van *et al.* achieve fast merging-based active clustering COBRA [17] via KMeans-based over-clustering. However, COBRA's use of sequential constraints makes it less scalable to the number of constraints. Gao *et al.* propose an efficient local search algorithm FastCCP [21] for semi-supervised clustering. However, the depth-first-search that FastCCP depends on can hardly handle super large datasets. Therefore, the lack of scalability is another major limitation of existing semi-supervised clustering methods.

To address the aforementioned issues, we propose Scalable Semi-supervised clustering via Structural Entropy, namely

Guangjie Zeng, Hao Peng, and Angsheng Li are with State Key Laboratory of Software Development Environment, Beihang University, Beijing 1000191, China. E-mail: {zengguangjie, penghao, angsheng}@buaa.edu.cn.

Jia Wu is with the Department of Computing, Macquarie University, Sydney NSW 2109, Australia. E-mail: jia.wu@mq.edu.au.

Chunyang Liu is with Didi Chuxing, Beijing 100193, China. E-mail: liuchunyang@didiglobal.com.

Philip S. Yu is with the Department of Computer Science, University of Illinois Chicago, Chicago, IL 60607, USA. E-mail: psyu@.uic.edu.

Manuscript received May 2024, revised August 2024, Accepted October 2024. (Corresponding author: Hao Peng.)

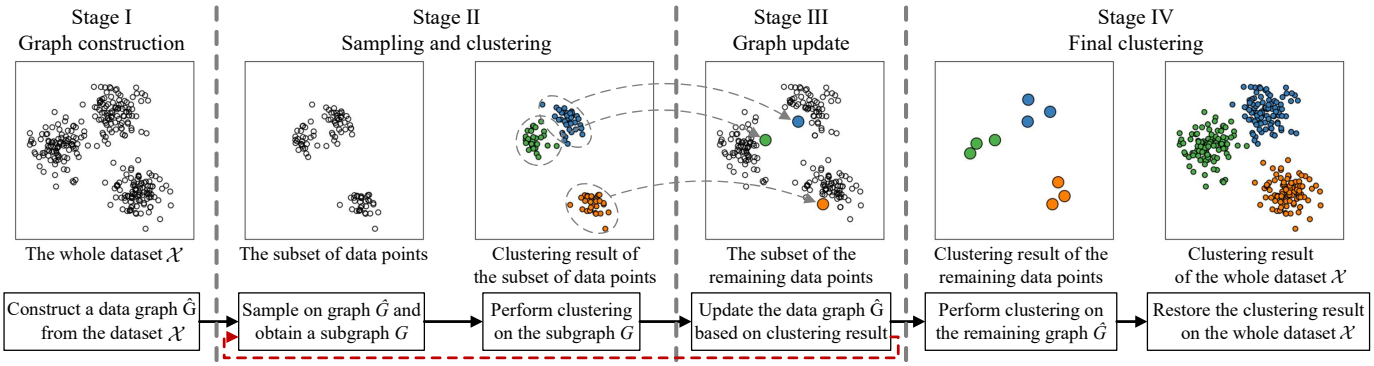

Fig. 1. An illustration of the sampling-based scalability strategy in SSSE framework.

SSSE, which, unlike deep learning-based methods, is fully explainable. Compared to existing semi-supervised clustering methods, SSSE is more scalable to the number of data points and constraints, and is more general for performing both partitioning and hierarchical clustering with different constraints. Specifically, we design objectives based on structural entropy along with a regularization term. This term is defined on a data graph G and a relation graph G', both sharing the same set of vertices, which represent the input data $\mathcal{X}$ and the prior information in the form of constraints, respectively. To achieve generalization on constraint types, we formulate different constraints as a uniform view and store them in the relation graph G'. To achieve scalability on data size, we adopt a sampling-based strategy following Cao *et al.* [22] and propose a neighborhood-preserving sampling strategy to alleviate the impedance of sampling. We devise the two-dimensional (2-D) SSSE and the high-dimensional (high-D) SSSE based on 2-D structural entropy and high-D structural entropy for semi-supervised partitioning and hierarchical clustering, respectively.

The detail of the sampling-based scalability strategy is shown in Fig. 1. Unlike Cao *et al.* [22], who sample subgraphs randomly, we adopt a neighborhood-preserving sampling strategy to maintain the intrinsic neighborhood structure of the original graph. Furthermore, we design our sampling process for parallel implementation, where the original graph is split into several subgraphs and then the clustering is performed on these subgraphs in parallel. This scalability strategy comprises four stages: graph construction, sampling and clustering, graph update, and final clustering. In the graph construction stage, we construct a data graph $\hat{G}$ from the whole dataset $\mathcal{X}$. In the sampling and clustering stage, we sample a subgraph G from $\hat{G}$ and perform clustering on G. In the graph update stage, we take clusters from the previous clustering as new data points, i.e., new vertices, and insert them back into $\hat{G}$. The procedure of Stage II and Stage III is performed repeatedly until the number of remaining vertices is small enough as the red arrow in Fig. 1 shows. In the final clustering stage, we perform clustering on the remaining graph $\hat{G}$ and restore the clustering result on the whole dataset $\mathcal{X}$. This sampling-based scalability strategy avoids direct clustering on the whole dataset $\mathcal{X}$ at a large scale. It reduces the data size after the sampling and clustering step, making SSSE an efficient and scalable method.

We comprehensively evaluate SSSE on 9 clustering datasets concerning two types of constraints. Compared to the unsupervised baseline SE, SSSE achieves up to 18% performance improvement and up to 30 times acceleration. We also apply SSSE to cluster cells in single-cell RNA-seq datasets at different scales, demonstrating the functionality of SSSE for biological data analysis. The main contributions of this paper can be summarized as follows:

- We propose a Scalable Semi-supervised clustering approach via Structural Entropy (SSSE). Its scalability is achieved by a sampling-based scalability strategy based on the neighborhood-preserving graph sampling strategy.
- We devise a unified approach for pairwise and label constraints and integrate them in a regularization term to form the objectives of SSSE.
- We design algorithms to effectively optimize the objectives of SSSE to enable semi-supervised partitioning and hierarchical clustering.
- We conduct experiments on real-world clustering datasets and single-cell RNA-seq datasets at different scales to verify the effectiveness and scalability of SSSE on semi-supervised clustering and biological data analysis.

This paper is an extension of our conference paper [23]. We have extended the technical contribution and the empirical evaluation of algorithms. The main extension in this paper can be summarized as follows: (1) We address the scalability issue of semi-supervised clustering. To achieve efficient and scalable semi-supervised clustering, we adopt a sampling-based scalability strategy based on the proposed neighborhood-preserving graph sampling strategy for SSSE. (2) We evaluate SSSE on much larger clustering datasets and also demonstrate the application of SSSE in biological data analysis at different scales. (3) We discuss more comprehensive related works, especially in scalable clustering.

II. Background and Preliminaries

A. Problem Formulation

Semi-supervised clustering aims to group data points from input data $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ into subsets, namely clusters, according to the given prior information in different forms of constraints. For partitioning clustering, data points are grouped into a certain non-overlapping partitioning $\mathcal{P} = \{X_1, X_2, ..., X_L\}$, where X_i is the i-th cluster and L is the total number of clusters. For hierarchical clustering, data points

are grouped into nested clusters organized as a clustering tree $\mathcal{T}$, where each cluster α on the tree is the union of its children. Semi-supervised clustering forms a partitioning or a clustering tree while integrating constraints to boost clustering accuracy and align with user preferences. In this work, we focus on two representative types of constraints: pairwise constraints, which reveal the relationship between data point pairs, and label constraints, which reveal the relationship between data points and ground truth class labels. Pairwise constraints include must-link constraints $M = \{(\mathbf{x}_i, \mathbf{x}_j)\colon l_i = l_j\}$, indicating that data point pair $(\mathbf{x}_i, \mathbf{x}_j)$ must belong to the same cluster, and cannot-link constraints $C = \{(\mathbf{x}_i, \mathbf{x}_j)\colon l_i \neq l_j\}$, indicating that data point pair $(\mathbf{x}_i, \mathbf{x}_j)$ must belong to different clusters, where l_i is the cluster indicator of $\mathbf{x}_i$. Label constraints include positive constraints $P = \{(\mathbf{x}_i, y_m)\colon \mathbf{x}_i \in y_m\}$, indicating that the true class label of $\mathbf{x}_i$ is y_m, and negative constraints $N = \{(\mathbf{x}_i, y_m)\colon \mathbf{x}_i \notin y_m\}$, indicating that the true class label of $\mathbf{x}_i$ is not y_m. We standardize these two types of constraints and store them in a relation graph G'. Together with a data graph G from the input data $\mathcal{X}$, we optimize objectives based on structural entropy to perform semi-supervised partitioning and hierarchical clustering.

B. Structural Entropy

In 2016, Li and Pan [24] proposed structural information theory and gave complete definitions of structural entropy. Intuitively, structural entropy is a measurement of graph complexity by encoding tree structures via characterizing the uncertainty of the hierarchical topology of graphs. The structural entropy of graph G is defined on an associated encoding tree $\mathcal{T}$, revealing the amount of uncertainty that remained in G after encoded by $\mathcal{T}$. Through structural entropy minimization, the optimized hierarchical clustering result of vertices in G is retained by $\mathcal{T}$. Structural entropy has been widely used in the field of bioinformatics [25], graph neural networks [26], and reinforcement learning [27]. It is also used in some other works related to clustering, including unsupervised hierarchical clustering [28], deep graph clustering [29], and our previous work SSE for semi-supervised clustering [23].

Let $G = (V, E, \mathbf{W})$ be an undirected weighted graph, where $V = \{v_1, ..., v_n\}$ is the vertex set, E is the edge set, and $\mathbf{W} \in \mathbb{R}^{n\times n}$ is the edge weight matrix. The encoding tree $\mathcal{T}$ of G is a hierarchical rooted tree where each tree node α associates with a vertex set T_α. The root node λ of $\mathcal{T}$ associates with $T_\lambda = V$ and each leaf node ν associates with T_ν containing only one vertex in V. For each non-leaf node $\alpha \in \mathcal{T}$, the successors of α are associated with disjoint vertex subsets, and the union of these subsets is T_α. The structural entropy of G given by $\mathcal{T}$ is defined as follows:

$$\mathcal{H}^{\mathcal{T}}(G) = \sum_{\alpha\in\mathcal{T},\alpha\neq\lambda} \mathcal{H}^{\mathcal{T}}(G;\alpha) = \sum_{\alpha\in\mathcal{T},\alpha\neq\lambda} -\frac{g_\alpha}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\alpha}{\mathcal{V}_{\alpha^-}}, \tag{1}$$

where $\mathcal{H}^{\mathcal{T}}(G;\alpha)$ is the assigned structural entropy of α, g_α is the cut, i.e., the sum of edge weights between vertices in and not in T_α, $\mathcal{V}_\alpha$ and $\mathcal{V}_G$ are the volumes, i.e., the sum of vertex degrees in T_α and G, respectively. The structural entropy of G is defined as:

$$\mathcal{H}(G) = \min_{\mathcal{T}}\{\mathcal{H}^{\mathcal{T}}(G)\}, \tag{2}$$

where $\mathcal{T}$ ranges over all possible encoding trees. The vertex sets associated with tree nodes form a hierarchical clustering of vertices in V.

K-D Structural entropy. The K-D structural entropy is the structural entropy given by the encoding trees with the height of at most K. When $K = 2$, the encoding tree represents graph partitioning, which can be used to perform partitioning clustering. A 2-D encoding tree $\mathcal{T}$ can be formulated as a graph partitioning $\mathcal{P} = \{X_1, X_2, ..., X_L\}$ of V, where X_i is a vertex subset called module associated with the i-th children of root λ. The structural entropy of G given by $\mathcal{P}$ is defined as:

$$\mathcal{H}^{\mathcal{P}}(G) = -\sum_{X\in\mathcal{P}}\sum_{v_i\in X}\frac{g_i}{\mathcal{V}_G}\log_2\frac{d_i}{\mathcal{V}_X} - \sum_{X\in\mathcal{P}}\frac{g_X}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_X}{\mathcal{V}_G}, \tag{3}$$

where d_i is the degree of vertex v_i, g_i is the cut, i.e., the sum of edge weights connecting v_i and other vertices, $\mathcal{V}_X$ and $\mathcal{V}_G$ are the volumes, i.e., the sum of vertex degrees in module X and graph G, respectively, and g_X is the cut, i.e., the sum of edge weights between vertices in and not in module X.

III. Methodology

In this section, we elaborate on the proposed scalable semi-supervised clustering via structural entropy (SSSE) in detail, which is based on our previous work SSE [23]. This paper addresses the scalability issue of semi-supervised clustering by graph sampling. The main idea of SSSE is to perform clustering on sampled subgraphs and insert obtained clusters back into the remaining graph as new data points. The framework of SSSE is depicted in Fig. 2.

For a dataset $\mathcal{X}$ along with given prior information, we first construct a data graph G and a relation graph G'. Then, we get a data subgraph and a relation subgraph by graph sampling and perform semi-supervised clustering on these two subgraphs. Clusters obtained in this round are aggregated into new data points and inserted back into G and G'. This sampling-and-clustering step is performed until the remaining graphs are small enough for fast clustering. Finally, we perform semi-supervised clustering on the remaining graphs and restore clusters from the previous steps.

Three key components of SSSE are graph construction (Sec. III-A), 2-D SSSE for semi-supervised partitioning clustering (Sec. III-B), and high-D SSSE for semi-supervised hierarchical clustering (Sec. III-C). In the graph construction component, we construct a p-nearest neighbor similarity graph from dataset $\mathcal{X}$ as the data graph G and a relation graph G' from the given prior information. In the 2-D SSSE component, we devise an objective function based on the 2-D structural entropy to achieve semi-supervised partitioning clustering and optimize it via two operators *merging* and *moving*. In the high-D SSSE component, we further extend the objective function for semi-supervised hierarchical clustering and optimize it via two operators *stretching* and *compressing*.

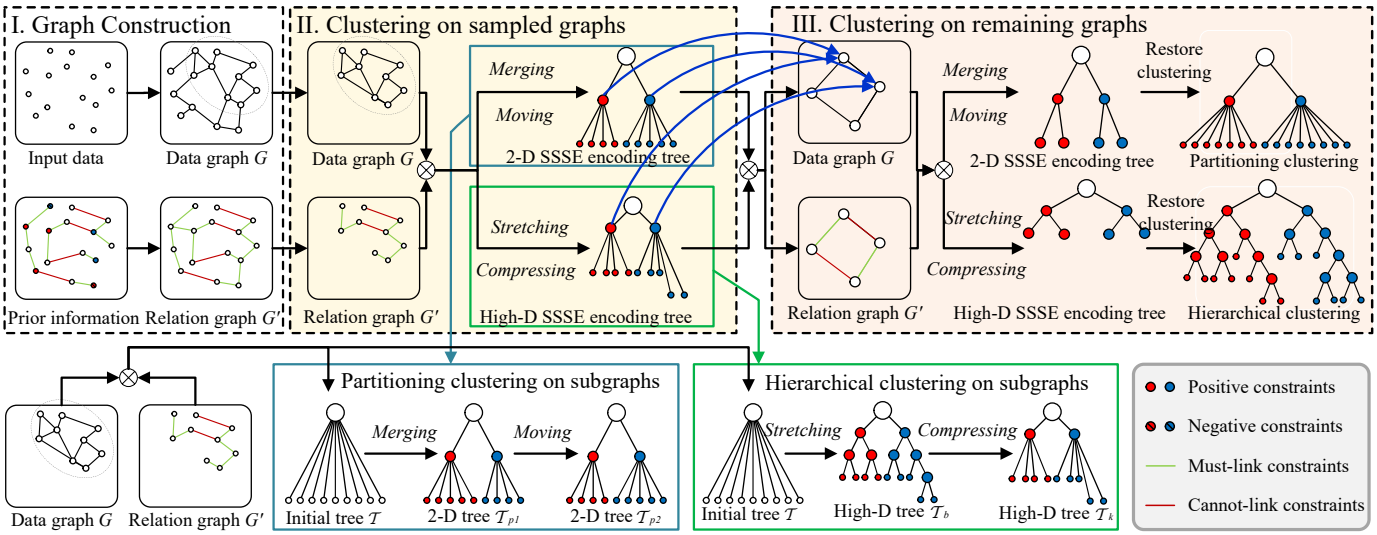

Fig. 2. The SSSE framework. We illustrate SSSE on a small dataset. (I) Two graphs G and G' are constructed from input data and prior information in the formats of pairwise constraints and label constraints, respectively. (II) Semi-supervised clustering is performed on the sampled graphs. (III) Semi-supervised clustering is performed on the remaining graphs, in which obtained clusters are taken as new data points. For semi-supervised partitioning clustering, clustering in (II) and (III) is achieved by two operators *merging* and *moving*, while for semi-supervised hierarchical clustering, clustering is achieved by *stretching* and *compressing*.

A. Graph Construction

The proposed SSSE requires a data graph G and a relation graph G' from the input data and prior information, respectively. The data graph G measures the intrinsic neighborhood structure of the input data $\mathcal{X}$, while the relation graph G' stores the information of constraints from the given prior information. We address the scalability issue by constructing p-nearest neighbor data graphs to avoid complex dense graphs. We also adopt a graph-sampling-based scalability strategy to avoid objective optimization on large graphs.

Data graph construction. Considering a dataset $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ with n data points, we construct a data graph $G = (V, E, \mathbf{W})$, where $V = \{v_1, v_2, ..., v_n\}$ is the set of vertices corresponding to data points in $\mathcal{X}$, E is the set of edges connecting vertices, and $\mathbf{W}$ is the set of edge weights measuring the similarities between data points. For two data points $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$, we measure their similarity $\mathbf{W}_{ij}$ by the radial basis function (RBF) kernel [30] or the cosine similarity according to the requirements of different tasks. In practice, we prefer a sparse data graph G which captures the manifold structures in high-dimensional spaces and alleviates the computational burden for the following clustering step. We sparsify G into a p-nearest neighbor graph by *NN-Descent* algorithm [31], an efficient algorithm for approximate nearest neighbor graph construction suitable for large-scale datasets.

Relation graph construction. Prior information is provided in the form of constraints for semi-supervised clustering. In this work, we give a uniform formulation of widely used pairwise and label constraints and store them in a relation graph. For a dataset $\mathcal{X}$ associated with data graph G, we construct a relation graph $G' = (V, E', \mathbf{W}')$ using the provided constraints, where G' shares the same vertex set with G.

Pairwise constraints consist of a set of must-link constraints $M = \{(\mathbf{x}_i, \mathbf{x}_j): l_i = l_j\}$ and a set of cannot-link constraints $C = \{(\mathbf{x}_i, \mathbf{x}_j): l_i \neq l_j\}$. If there exists a pair of data points $(\mathbf{x}_i, \mathbf{x}_j) \in M$, an edge exists in E' with a positive value γ_M added to the edge weight $\mathbf{W}'_{ij}$. If there exists a pair of data points $(\mathbf{x}_i, \mathbf{x}_j) \in C$, an edge exists in E' with a negative value γ_C added to the edge weight $\mathbf{W}'_{ij}$. The parameters γ_M and γ_C control the role of constraints and are set following Bai *et al.* [3]. For a pair of data points $(\mathbf{x}_i, \mathbf{x}_j)$ with similarity $\mathbf{W}_{ij}$, we set $\gamma_M = \max(\mathbf{W}) - \mathbf{W}_{ij}$, where $\max(\mathbf{W})$ is the maximum similarity among all pairs of data points. To balance the influence of positive values and negative values in G' when the number of must-link constraints and cannot-link constraints is not the same, we set $\gamma_C = \rho(\min(\mathbf{W}) - \mathbf{W}_{ij})$, where ρ is the ratio between the number of positive values and negative values in G'.

Label constraints consist of a set of positive constraints $P = \{(\mathbf{x}_i, y_m): \mathbf{x}_i \in y_m\}$ and a set of negative constraints $N = \{(\mathbf{x}_i, y_m): \mathbf{x}_i \notin y_m\}$. To form a uniform representation of constraints, we convert label constraints into pairwise constraints which are more compatible for SSSE. For two data points $\mathbf{x}_i$ and $\mathbf{x}_j$, the conversion rules are set as follows: (1) If they have positive constraints with the same label, an edge exists in E' with a positive value γ_M added to the edge weight $\mathbf{W}'_{ij}$. (2) If they have positive constraints with different labels, an edge exists in E' with a negative value γ_C added to the edge weight $\mathbf{W}'_{ij}$. (3) If they have positive constraint and negative constraint respectively with the same class label, the first data point belongs to this class while the second does not. In that case, an edge exists in E' with a negative value γ_C added to the edge weight $\mathbf{W}'_{ij}$. Label constraints conversion tends to generate a dense relation graph G', where vertices with label constraints connect to all other vertices with label constraints. To avoid a complex dense relation graph, we only retain two positive and two negative edges at most for each vertex in G' when performing constraints conversion.

Neighborhood-preserving graph sampling strategy. To achieve scalable semi-supervised clustering, we adopt a sampling-based scalability strategy (Fig. 1) that avoids SSSE from objective optimization on large graphs. After constructing

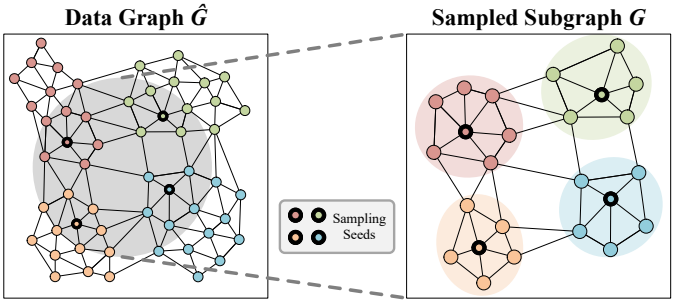

Fig. 3. Neighborhood-preserving graph sampling strategy. We randomly select a set of seed vertices and extend the set according to the neighbor vertices of its belonged vertices.

the data graph, we sample it into a set of subgraphs and perform semi-supervised clustering on these subgraphs. Clusters from subgraphs shrink into data points, which decreases the size of the graph. This graph sampling step improves the scalability of SSSE. However, a simple random graph sampling will break the nearest-neighbor structure in the data graph thus impeding the performance of SSSE. Motivated by previous research [32] that notices the importance of the local neighborhood structure of graphs, we propose a neighborhood-preserving graph sampling strategy, as shown in Fig. 3. This sampling strategy produces smaller subgraphs and preserves the intrinsic neighborhood structure of graphs at the same time. The experimental results in Fig. 4 show that our proposed sampling strategy significantly outperforms random graph sampling.

Specifically, given a whole data graph $\hat{G}$ with an associated relation graph $\hat{G}'$, we randomly select n_{seed} sampling center vertices as the sampling seeds stored in the current selected vertex set S_{select}. For each selected vertex, we extend to its p nearest neighbor vertices according to $\hat{G}$. This extension step is conducted until the size of S_{select} reaches a sampling size threshold n_s. We sample a list of vertex sets without replacement, and the remaining vertices in $\hat{G}$ are allocated to the final vertex set. Each vertex set corresponds to a data subgraph G and a relation subgraph G' from $\hat{G}$ and $\hat{G}'$, respectively. These subgraphs are utilized to form clusters via objective optimization. There are chances that conflicts exist between constraints in $\hat{G}'$ and the graph sampling output. For instance, two vertices with a must-link constraint might be sampled into different subgraphs. This makes constraints ineffective in the current round. However, these constraints can become effective in the following rounds when $\hat{G}'$ is updated and vertices with constraints are sampled into the same subgraph.

B. 2-D SSSE

The 2-D structural entropy of a graph represents the lowest possible structural entropy achievable through graph partitioning. Consequently, it can be effectively utilized for partitioning clustering. In our previous work [23], we propose 2-D SSE based on 2-D structural entropy for semi-supervised partitioning clustering. In this subsection, we further address the scalability issue and present 2-D SSSE for scalable semi-supervised partitioning clustering. We achieve this by proposing a 2-D structural entropy-based objective function and the corresponding scalable optimization algorithm.

Given a dataset $\mathcal{X}$ at the size of n, we construct a data graph $\hat{G}$ and a relation graph $\hat{G}'$, and sample $\hat{G}$ into a list of subgraphs as described in Section III-A. We perform clustering on these subgraphs and insert the obtained clusters as new data points back into $\hat{G}$ and $\hat{G}'$. Suppose we obtain a data subgraph G containing n_s vertices along with a relation subgraph G' via graph sampling, we intend to perform semi-supervised partitioning clustering on these two graphs. To integrate the given constraints into the clustering procedure, we propose a regularization term defined on these two graphs. The objective function of 2-D SSSE on G and G' is defined as follows:

$$\mathcal{L}^{\mathcal{P}}(G, G') = \mathcal{H}^{\mathcal{P}}(G) + \phi \mathcal{E}^{\mathcal{P}}(G, G'), \tag{4}$$

where $\mathcal{H}^{\mathcal{P}}(G)$ is the 2-D structural entropy defined in Eq. (3) and $\mathcal{E}^{\mathcal{P}}(G, G')$ is the regularization term for constraints violation, which is defined as follows:

$$\mathcal{E}^{\mathcal{P}}(G, G') = -\sum_{X \in \mathcal{P}} \frac{g'_X}{\mathcal{V}_G} \log_2 \frac{\mathcal{V}_X}{\mathcal{V}_G}, \tag{5}$$

where g'_X is the sum of edge weights in G' between vertices in and not in partition X, and other notations share the same meaning with notations in Eq. (3).

The intuition of the regularization term is that we modify g_X, *i.e.*, the cut of module X in Eq. (3) according to the constraints, which is increased when must-link constraints are violated, and decreased when cannot-link constraints are satisfied. A positive value of $\mathbf{W}'_{ij}$ in G' means v_i and v_j should belong to the same partition, and $\mathcal{E}^{\mathcal{P}} > 0$ if they are not, leading to a penalty added to $\mathcal{L}^{\mathcal{P}}$. A negative value of $\mathbf{W}'_{ij}$ in G' means v_i and v_j should belong to different partition, and $\mathcal{E}^{\mathcal{P}} < 0$ if they are, leading to a reward added to $\mathcal{L}^{\mathcal{P}}$. When no constraint exists, i.e., $\mathcal{E}^{\mathcal{P}} = 0$, we only minimize unsupervised 2-D structural entropy. In all, $\mathcal{E}^{\mathcal{P}}$ penalizes modules that violate must-link constraints and rewards modules that satisfy cannot-link constraints.

Minimizing 2-D SSSE objective. Given a data subgraph G and a relation subgraph G', we minimize 2-D SSSE objective defined by Eq. (4) via two operators *merging* [24] and *moving* on the encoding tree $\mathcal{T}$. The operator *merging* seeks to union vertex subsets of two sister nodes and delete one node. For two sister nodes $\alpha, \beta \in \mathcal{T}$ with associated vertex subsets X and Y, the operator *merging* is defined as: (1) set $X = X \cup Y$, (2) delete β. The decrease amount of $\mathcal{L}^{\mathcal{P}}(G, G')$ is given by:

$$\begin{aligned} \Delta\mathcal{L}^{\mathcal{M}}_{X,Y} = \frac{1}{\mathcal{V}_G}[&(\mathcal{V}_X - g_X - g'_X)\log_2 \mathcal{V}_X \\ &+ (\mathcal{V}_Y - g_Y - g'_Y)\log_2 \mathcal{V}_Y \\ &- (\mathcal{V}_{X\cup Y} - g_{X\cup Y} - g'_{X\cup Y})\log_2 \mathcal{V}_{X\cup Y} \\ &+ (g_X + g_Y - g_{X\cup Y} + g'_X + g'_Y - g'_{X\cup Y})\log_2 \mathcal{V}_G], \end{aligned} \tag{6}$$

where $\mathcal{M}$ denotes the operator *merging* , $\mathcal{V}_X$ is the volume of X in G, $\mathcal{V}_G$ is the volume of G, g_X and g'_X are the cuts of X in G and G', respectively. The operator *moving* aims to find a better module for a vertex by moving the vertex from one module to another. For a node $\alpha \in \mathcal{T}$ with associated module X and a vertex $v_i \in X$, a target node $\beta \in \mathcal{T}$ with associated

Algorithm 1 Algorithm to Minimize Objective Eq. (4)

Input: $G = (V, E, \mathbf{W})$, $G' = (V, E', \mathbf{W}')$
Output: Encoding tree $\mathcal{T}$ and partitioning $\mathcal{P}$

Initialize $\mathcal{T}$ containing all vertices as tree leaves
// *Merging* stage
repeat
  Merge a chosen module pair (X, Y) into $X \cup Y$ condition on $\arg\max_{X,Y}\{\Delta\mathcal{L}^{\mathcal{M}}_{X,Y}\}$ via Eq. (6)
  Update $\Delta\mathcal{L}^{\mathcal{M}}$ for module pairs connected to X or Y
until $\Delta\mathcal{L}^{\mathcal{M}} < 0$ for all module pairs
// *Moving* stage
repeat
  for each vertex $v_i \in V$ **do**
    Remove vertex v_i from the original module X
    Insert node v_i into the module Y condition on $\arg\max_Y\{\Delta\mathcal{L}^{\mathcal{R}}_{X,v_i} - \Delta\mathcal{L}^{\mathcal{I}}_{Y,v_i}\}$ via Eqs. (7) and (8)
  end for
until $\mathcal{L}^{\mathcal{P}}(G, G')$ does not decrease in current iteration

module Y is found, where Y connects to v_i by edges in G, the *moving* operator is defined as: (1) remove v_i from X, (2) insert v_i from Y. The decrease amount of $\mathcal{L}^{\mathcal{P}}(G, G')$ by removing v_i from X is given by:

$$\Delta\mathcal{L}^{\mathcal{R}}_{X,v_i} = \frac{\mathcal{V}_X - g_X - g'_X}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_X}{\mathcal{V}_G} - \frac{\mathcal{V}_{X\setminus\{v_i\}} - g_{X\setminus\{v_i\}} - g'_{X\setminus\{v_i\}}}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_{X\setminus\{v_i\}}}{\mathcal{V}_G}, \tag{7}$$

where $\mathcal{R}$ denotes vertex removing and $X\setminus\{v_i\}$ denotes removing vertex v_i from X. The increase amount of $\mathcal{L}^{\mathcal{P}}(G, G')$ by inserting v_i into Y is given by:

$$\Delta\mathcal{L}^{\mathcal{I}}_{Y,v_i} = -\frac{\mathcal{V}_Y - g_Y - g'_Y}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_Y}{\mathcal{V}_G} + \frac{\mathcal{V}_{Y\cup\{v_i\}} - g_{Y\cup\{v_i\}} - g'_{Y\cup\{v_i\}}}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_{Y\cup\{v_i\}}}{\mathcal{V}_G}, \tag{8}$$

where $\mathcal{I}$ denotes vertex inserting and $Y \cup \{v_i\}$ denotes inserting v_i into Y. We initialize $\mathcal{T}$ to contain a root node λ and n leaves where each leaf is associated with one vertex in G, and then sequentially apply *merging* and *moving* operators until convergence. The optimization procedure is summarized in Algorithm 1.

Complexity analysis. For a subgraph G with n_s vertices, the time complexity of *merging* stage is $O(n_s\log^2 n_s)$ [24]. In the *moving* stage, each iteration requires calculating $\Delta\mathcal{L}^{\mathcal{R}}_{X,v_i}$ and $\Delta\mathcal{L}^{\mathcal{I}}_{Y,v_i}$ for every vertex v_i and every possible module Y, which takes the time of $O(n_s l)$. Taken together, the time complexity of Algorithm 1 is $O(n_s\log^2 n_s + n_s l t)$, where l and t denote the number of vertices, clusters, and iterations respectively. For a dataset $\mathcal{X}$ with n data points and l classes, the whole data graph will be separated into $\lfloor n/n_s \rfloor$ subgraphs, which will give $\lceil nl/n_s \rceil$ vertices in the remaining data graph after a round of clustering. The remaining graph will be separated into $\lfloor nl/n_s^2 \rfloor$ subgraphs. In all, 2-D SSSE requires running Algorithm 1 $\lfloor n/(n_s - l) \rfloor$ times, and the total time complexity is $O((nn_s\log^2 n_s + nn_s lt)/(n_s - l))$. Considering that $n_s \gg l$ for most datasets, the time complexity of 2-D SSSE is $O(n\log^2 n_s + nlt)$, which is approximately linear to the number of data points.

C. High-D SSSE

The high-D structural entropy of a graph represents the lowest possible structural entropy given by a hierarchical encoding tree. Hierarchical clustering can be achieved by high-D structural entropy minimization [28]. In our previous work [23], we propose high-D SSE based on high-D structural entropy for semi-supervised hierarchical clustering. In this subsection, we present the high-D SSSE for scalable hierarchical clustering via the graph sampling-based scalability strategy. This is achieved by extending the objective function of the 2-D SSSE and proposing the corresponding scalable optimization algorithm.

Given a dataset $\mathcal{X}$ along with the constructed data graph $\hat{G}$ and relation graph $\hat{G}'$, we sample $\hat{G}$ into a list of subgraphs as described in Section III-A. We apply the *stretching* operator to achieve binary hierarchical clustering and then apply the *compressing* operator to obtain clusters. These clusters are inserted back into $\hat{G}$ and $\hat{G}'$ as new data points for the following hierarchical clustering. For a data subgraph G containing n_s vertices and the corresponding relation subgraph G' via graph sampling, we intend to perform semi-supervised hierarchical clustering on these two graphs. Following the objective of 2-D SSSE defined in Eq (4) where the constraints are applied on all modules, the high-D SSSE objective applies constraints on all internal tree nodes in $\mathcal{T}$, i.e., tree nodes except for the root node and tree leaves in $\mathcal{T}$. The objective function is defined as follows:

$$\mathcal{L}^{\mathcal{T}}(G, G') = \mathcal{H}^{\mathcal{T}}(G) + \phi\mathcal{E}^{\mathcal{T}}(G, G'), \tag{9}$$

where $\mathcal{E}^{\mathcal{T}}(G, G')$ is a regularization term for constraints violation, and it is defined as:

$$\mathcal{E}^{\mathcal{T}}(G, G') = \sum_{\alpha\in\mathcal{T}, 1<|T(\alpha)|<|V|} -\frac{g'_\alpha}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\alpha}{\mathcal{V}_{\alpha^-}}, \tag{10}$$

where g'_α is the cut of α in G', $|T(\alpha)|$ is the number of vertices in subset $T(\alpha)$ associating to α, and other notations share the same meaning with notations in Eq. (1). For each internal node in $\mathcal{T}$, the penalty term penalizes the violation of must-link constraints and rewards the satisfaction of cannot-link constraints.

Minimizing high-D SSSE objective. Given a data subgraph G and a relation subgraph G', we minimize high-D SSSE objective defined by Eq. (9) via two operators *stretching* and *compressing* on the encoding tree $\mathcal{T}$ [28]. The operator *stretching* seeks to stretch the encoding tree by adding a parent node above a pair of sister nodes. For a pair of sister nodes $(\alpha, \beta) \in \mathcal{T}$ whose parent is γ, node *stretching* is defined as inserting a new node δ between γ and (α, β), i.e., (1) set $\alpha^- = \delta$, (2) set $\beta^- = \delta$, (3) set $\delta^- = \gamma$. The decrease amount of $\mathcal{L}^{\mathcal{T}}(G, G')$ is given by:

$$\Delta\mathcal{L}^{\mathcal{S}}_{\alpha,\beta} = \frac{g_\alpha + g_\beta - g_\delta + g'_\alpha + g'_\beta - g'_\delta}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\gamma}{\mathcal{V}_\delta}, \tag{11}$$

Algorithm 2 Algorithm to Minimize Objective Eq. (9)

Input: $G = (V, E, \mathbf{W})$, $G' = (V, E', \mathbf{W}')$, height K
Output: Binary tree $\mathcal{T}_b$ and height K tree $\mathcal{T}_K$

Initialize $\mathcal{T}$ with a root node λ and all vertices as tree leaves
// *Stretching* stage
repeat
  Stretch a chosen node pair $\{\alpha, \beta\}$ condition on $\arg\max_{\alpha,\beta}\{\Delta\mathcal{L}^{\mathcal{S}}_{\alpha,\beta}\}$ via Eq. (11)
  Update $\Delta\mathcal{L}^{\mathcal{S}}$ for node pairs connected to α or β
until The children number of λ is two, resulting in binary tree $\mathcal{T}_b$
// *Compressing* stage
repeat
  Remove a chosen tree node $\alpha \in \mathcal{T}$ condition on $\arg\max_{\alpha}\{\Delta\mathcal{L}^{\mathcal{C}}_{\alpha}\}$ via Eq. (12)
until Height of encoding tree $\mathcal{T}$ is not larger than K, resulting in $\mathcal{T}_K$

where $\mathcal{S}$ denotes node *stretching* . Applying *stretching* on the initial encoding tree iteratively results in a binary encoding tree $\mathcal{T}_b$. The operator *compressing* aims to restrict the height of the encoding tree by deleting certain internal tree nodes. For a node $\alpha \in \mathcal{T}$ contains a set of children $\{\beta_1, ..., \beta_m\}$ and its parent is γ, node *compressing* is defined as: (1) remove node α, (2) for each child node β_i of α, set $\beta_i^- = \gamma$. The decrease amount of $\mathcal{L}^{\mathcal{T}}(G, G')$ is given by:

$$\Delta\mathcal{L}^{\mathcal{C}}_{\alpha} = \frac{\sum\limits_{i} g_{\beta_i} + \sum\limits_{|T(\beta_i)|>1} g'_{\beta_i} - g_{\alpha} - g'_{\alpha}}{\mathcal{V}_G} \log_2 \frac{\mathcal{V}_{\alpha}}{\mathcal{V}_{\gamma}}, \quad (12)$$

where $\mathcal{C}$ denotes node *compressing* . Applying *compressing* on the binary encoding tree results in a multinary encoding tree. By restricting the height of the encoding tree to be less than the required height K, we can obtain the K-D encoding tree. We summarize this optimization procedure in Algorithm 2. We apply $\mathcal{C}ompressing$ until $K = 2$, where the clusters shrink into new vertices for the following graph sampling step. For a graph G with n_s vertices and m_s edges, the time complexity of Algorithm 2 is $O(h_{\max}(m_s log n_s + n_s))$, where h_{max} is the height of $\mathcal{T}_b$. For a dataset $\mathcal{X}$ with n data points and l classes, high-d SSSE requires running Algorithm 2 $\lfloor n/(n_s - l) \rfloor$ times, and the total time complexity is $O(nh_{\max}(m_s log n_s + n_s)/(n_s - l))$.

D. Limitation

Although the graph sampling-based scalability strategy provides SSSE the ability to achieve efficient clustering on datasets at large scales, this strategy inevitably brings in the loss of clustering accuracy. Despite the proposed neighborhood-preserving graph sampling strategy better preserves the intrinsic neighborhood structure of graphs which alleviates the loss of clustering accuracy, minimizing Eq. (4) and Eq. (9) on subgraphs rather than the whole graphs leads to local optimal clustering, and the performance varies with different sampling results.

TABLE I
DESCRIPTION OF DATASETS AT DIFFERENT SCALES.

Scale	Dataset	#Data points	#Features	#Classes
Small	Yale [33]	165	1024	15
	Wine [34]	178	13	3
	ORL [33]	400	1024	40
	Australian [34]	690	14	2
	Isolet [33]	1,560	617	26
Medium	COIL100 [33]	7,200	7200	100
	USPS [33]	9,298	256	10
Large	MNIST [33]	70,000	784	10
	EMNIST [35]	131,600	784	47

IV. EXPERIMENTS

Our proposed SSSE algorithm is capable of tackling both semi-supervised partitioning and hierarchical clustering. Regarding the evaluation of both tasks, we design two groups of experiments, in which we compare SSSE against baselines for semi-supervised partitioning clustering (Sec. IV-A) and semi-supervised hierarchical clustering (Sec. IV-B). We conduct experiments on 9 widely used real-world clustering datasets divided into three groups: small datasets (Yale, Wine, ORL, Australian, Isolet), Medium datasets (COIL100, USPS), and Large datasets (MNIST, EMNIST). The details of the above datasets are summarized in Table I.

A. Semi-Supervised Partitioning Clustering

In this part, we evaluate the performance of SSSE on semi-supervised partitioning clustering. We adopt three commonly used clustering metrics, Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), and Clustering Accuracy (ACC), for performance evaluation. We compare SSSE with several unsupervised clustering methods (GBSC [36], U-SPEC [37], U-SENC [37], SE [24]), several semi-supervised clustering methods with pairwise constraints (PCPSNMF [38], OneStepPCP [39], CMS [9]), several semi-supervised clustering methods with label constraints (Seeded-KMeans [1], S4NMF [40] , and one method with both pairwise and label constraints (SC-MPI [3]).

Implementation Details. For a given dataset $\mathcal{X}$ with n data points divided into k clusters according to the ground truth, we construct the data graph G as a p-nearest neighbor graph and set p as $\lfloor 20k/\log_2^2 n \rfloor + 1$, since the number of clusters by minimizing $\mathcal{H}^{\mathcal{P}}$ is approximate $\Theta(p \log_2^2 n)$ [24]. The similarity measure used for graph construction is the RBF kernel-based similarity, and the kernel width is empirically set to be $\sigma^2 = 50$. The number of sampling seeds is empirically set to be $n_{seed} = 10$ and the sampling size is empirically set to be $n_s = 1000$. For medium and large datasets utilizing graph sampling, we set p as $\lfloor 20k/\log_2^2 n_s \rfloor + 1$ since the objective optimization is performed on sampled subgraphs. We generate constraints using the ground truth class labels from each dataset. For the group of methods with pairwise constraints, we set the number of must-link constraints the same as cannot-link constraints to be $0.2n$. For the group of methods with label constraints, we set the number of positive constraints the same as negative constraints to be $0.1n$. The

TABLE II
PERFORMANCE OF METHODS FOR SEMI-SUPERVISED PARTITIONING CLUSTERING ON CLUSTERING DATASETS. **BOLD**: THE BEST PERFORMANCE ON EACH GROUP OF METHODS. OOM: OUT-OF-MEMORY ERROR. N/A: RUNNING TIME LONGER THAN 2 HOURS.

Metric	Cat.	Method(%)	Yale	ORL	Isolet	COIL100	USPS	MNIST	EMNIST
ARI	Unsup.	GBSC	12.30$_{\pm 1.47}$	09.24$_{\pm 02.63}$	15.57$_{\pm 07.67}$	23.85$_{\pm 0.53}$	28.01$_{\pm 0.05}$	22.31$_{\pm 1.87}$	OOM
		U-SPEC	29.51$_{\pm 1.50}$	54.70$_{\pm 1.03}$	55.41$_{\pm 2.66}$	52.91$_{\pm 1.63}$	64.67$_{\pm 0.28}$	58.83$_{\pm 3.79}$	22.90$_{\pm 0.51}$
		U-SENC	31.89$_{\pm 2.73}$	54.24$_{\pm 1.45}$	57.42$_{\pm 1.85}$	51.82$_{\pm 1.29}$	73.11$_{\pm 5.92}$	68.07$_{\pm 1.98}$	26.33$_{\pm 0.42}$
		SE	34.32$_{\pm 0.00}$	53.56$_{\pm 0.00}$	59.13$_{\pm 0.00}$	58.75$_{\pm 0.00}$	65.28$_{\pm 0.00}$	N/A	N/A
	Pairwise	PCPSNMF	26.03$_{\pm 3.19}$	50.47$_{\pm 4.49}$	38.75$_{\pm 7.69}$	OOM	OOM	OOM	OOM
		OneStepPCP	25.45$_{\pm 0.84}$	39.81$_{\pm 3.46}$	50.86$_{\pm 2.10}$	23.61$_{\pm 1.88}$	68.43$_{\pm 4.12}$	N/A	N/A
		CMS	06.60$_{\pm 2.12}$	28.34$_{\pm 5.16}$	49.15$_{\pm 1.59}$	29.07$_{\pm 1.64}$	72.49$_{\pm 2.95}$	N/A	N/A
		SC-MPI	31.39$_{\pm 2.30}$	52.53$_{\pm 6.14}$	46.06$_{\pm 4.77}$	56.75$_{\pm 2.50}$	21.70$_{\pm 4.84}$	OOM	OOM
		SSSE (Ours)	**40.68**$_{\pm 4.11}$	**56.28**$_{\pm 1.16}$	**63.31**$_{\pm 1.30}$	**70.94**$_{\pm 0.44}$	69.95$_{\pm 1.32}$	**74.55**$_{\pm 3.41}$	25.35$_{\pm 1.62}$
	Label	Seeded KMeans	24.30$_{\pm 2.22}$	46.13$_{\pm 2.62}$	**64.88**$_{\pm 1.70}$	54.69$_{\pm 1.59}$	57.22$_{\pm 0.05}$	40.97$_{\pm 2.01}$	17.61$_{\pm 0.21}$
		S4NMF	24.49$_{\pm 0.81}$	47.05$_{\pm 1.66}$	58.79$_{\pm 1.61}$	49.89$_{\pm 0.51}$	66.58$_{\pm 4.89}$	OOM	OOM
		SC-MPI	22.37$_{\pm 5.03}$	26.00$_{\pm 2.42}$	63.46$_{\pm 1.32}$	**73.09**$_{\pm 1.13}$	**90.41**$_{\pm 0.66}$	OOM	OOM
		SSSE (Ours)	**34.61**$_{\pm 2.05}$	53.99$_{\pm 0.93}$	60.71$_{\pm 0.76}$	66.64$_{\pm 1.13}$	68.97$_{\pm 2.67}$	**76.07**$_{\pm 2.07}$	24.97$_{\pm 1.65}$
NMI	Unsup.	GBSC	38.34$_{\pm 2.10}$	48.72$_{\pm 3.88}$	43.63$_{\pm 0.74}$	62.61$_{\pm 0.23}$	43.45$_{\pm 0.09}$	36.07$_{\pm 2.28}$	OOM
		U-SPEC	55.61$_{\pm 1.29}$	83.28$_{\pm 0.20}$	79.35$_{\pm 1.33}$	80.57$_{\pm 0.52}$	77.84$_{\pm 0.26}$	68.84$_{\pm 1.74}$	46.66$_{\pm 0.31}$
		U-SENC	57.32$_{\pm 1.65}$	83.23$_{\pm 0.67}$	79.21$_{\pm 0.90}$	78.80$_{\pm 0.49}$	82.02$_{\pm 1.90}$	75.79$_{\pm 1.26}$	50.47$_{\pm 0.48}$
		SE	60.87$_{\pm 0.00}$	86.38$_{\pm 0.00}$	82.74$_{\pm 0.00}$	87.79$_{\pm 0.00}$	73.67$_{\pm 0.00}$	N/A	N/A
	Pairwise	PCPSNMF	55.36$_{\pm 1.58}$	81.42$_{\pm 1.73}$	68.72$_{\pm 4.23}$	OOM	OOM	OOM	OOM
		OneStepPCP	52.05$_{\pm 0.80}$	77.83$_{\pm 0.73}$	76.85$_{\pm 1.17}$	71.81$_{\pm 0.80}$	**81.75**$_{\pm 0.70}$	N/A	N/A
		CMS	34.37$_{\pm 3.04}$	73.30$_{\pm 3.01}$	77.07$_{\pm 1.01}$	70.42$_{\pm 0.44}$	78.26$_{\pm 1.21}$	N/A	N/A
		SC-MPI	60.15$_{\pm 1.82}$	83.45$_{\pm 1.95}$	72.47$_{\pm 2.84}$	85.40$_{\pm 1.03}$	37.56$_{\pm 5.72}$	OOM	OOM
		SSSE (Ours)	**65.41**$_{\pm 2.21}$	**87.00**$_{\pm 0.34}$	**83.36**$_{\pm 0.79}$	**91.01**$_{\pm 0.11}$	77.61$_{\pm 0.83}$	**77.59**$_{\pm 1.60}$	**50.61**$_{\pm 0.63}$
	Label	Seeded KMeans	51.36$_{\pm 1.98}$	78.39$_{\pm 1.11}$	80.29$_{\pm 0.93}$	80.30$_{\pm 0.54}$	63.57$_{\pm 0.04}$	51.69$_{\pm 1.88}$	41.28$_{\pm 0.15}$
		S4NMF	50.16$_{\pm 0.98}$	76.95$_{\pm 0.98}$	78.18$_{\pm 0.93}$	78.45$_{\pm 0.25}$	74.73$_{\pm 3.23}$	OOM	OOM
		SC-MPI	51.91$_{\pm 3.59}$	69.97$_{\pm 1.27}$	80.00$_{\pm 0.56}$	89.23$_{\pm 0.43}$	**88.01**$_{\pm 0.66}$	OOM	OOM
		SSSE (Ours)	**61.52**$_{\pm 1.45}$	**86.47**$_{\pm 0.25}$	**83.08**$_{\pm 0.32}$	**89.94**$_{\pm 0.23}$	76.95$_{\pm 1.26}$	**78.19**$_{\pm 1.03}$	**50.67**$_{\pm 0.71}$
ACC	Unsup.	GBSC	12.63$_{\pm 1.21}$	09.75$_{\pm 2.03}$	14.70$_{\pm 1.00}$	23.72$_{\pm 0.39}$	27.07$_{\pm 2.77}$	22.95$_{\pm 0.08}$	OOM
		U-SPEC	48.12$_{\pm 2.29}$	66.15$_{\pm 1.53}$	62.59$_{\pm 2.16}$	58.25$_{\pm 2.21}$	67.68$_{\pm 4.85}$	69.84$_{\pm 2.53}$	37.84$_{\pm 1.17}$
		U-SENC	50.49$_{\pm 2.97}$	67.18$_{\pm 1.45}$	63.85$_{\pm 2.14}$	56.97$_{\pm 0.97}$	77.90$_{\pm 5.75}$	75.97$_{\pm 2.86}$	41.15$_{\pm 0.71}$
		SE	57.58$_{\pm 0.00}$	54.00$_{\pm 0.00}$	61.73$_{\pm 0.00}$	67.68$_{\pm 0.00}$	60.62$_{\pm 0.00}$	N/A	N/A
	Pairwise	PCPSNMF	49.70$_{\pm 3.78}$	63.68$_{\pm 3.51}$	51.17$_{\pm 2.65}$	OOM	OOM	OOM	OOM
		OneStepPCP	47.33$_{\pm 1.75}$	57.28$_{\pm 2.55}$	57.50$_{\pm 2.17}$	38.84$_{\pm 1.96}$	66.90$_{\pm 2.15}$	N/A	N/A
		CMS	28.67$_{\pm 2.36}$	49.35$_{\pm 1.89}$	53.67$_{\pm 2.36}$	33.20$_{\pm 1.74}$	75.67$_{\pm 1.54}$	N/A	N/A
		SC-MPI	50.73$_{\pm 5.17}$	66.43$_{\pm 4.52}$	55.12$_{\pm 7.37}$	59.56$_{\pm 2.59}$	37.22$_{\pm 2.63}$	OOM	OOM
		SSSE (Ours)	**59.88**$_{\pm 2.23}$	**69.33**$_{\pm 0.87}$	**64.67**$_{\pm 1.96}$	**76.65**$_{\pm 1.31}$	**78.66**$_{\pm 2.47}$	**84.32**$_{\pm 3.87}$	**41.54**$_{\pm 1.64}$
	Label	Seeded KMeans	45.76$_{\pm 3.58}$	62.53$_{\pm 1.92}$	74.63$_{\pm 1.88}$	62.14$_{\pm 0.83}$	75.00$_{\pm 0.03}$	59.93$_{\pm 1.71}$	33.35$_{0.43}$
		S4NMF	43.77$_{\pm 1.59}$	60.45$_{\pm 1.70}$	68.89$_{\pm 1.22}$	56.65$_{\pm 0.75}$	72.16$_{\pm 2.56}$	OOM	OOM
		SC-MPI	46.42$_{\pm 2.36}$	48.88$_{\pm 2.32}$	**76.92**$_{\pm 1.18}$	**83.53**$_{\pm 0.83}$	**95.02**$_{\pm 0.29}$	OOM	OOM
		SSSE (Ours)	**57.94**$_{\pm 2.77}$	**68.38**$_{\pm 0.98}$	62.53$_{\pm 1.51}$	76.02$_{\pm 1.32}$	78.75$_{\pm 2.09}$	**82.48**$_{\pm 3.10}$	**41.85**$_{\pm 1.34}$

parameter ϕ in Eq. (4) is simply set as $\phi = 1$. For all baselines, we obtain the source codes from the official implementation in the author's websites. The parameters are set according to the original paper or the official implementations. The experiments are conducted 10 times on one server with two Intel(R) 2.30 GHz CPUs and 500 GB RAM. More detailed information about the experimental setup can be found in the Appendix. The codes for all baseline models and SSSE, along with all datasets, are publicly accessible on GitHub[1].

Experimental Results. In Table II, we show the performance of methods for partitioning clustering on 7 real-world clustering datasets. We compare SSSE with three groups of methods, i.e., the unsupervised clustering group, the semi-supervised clustering with pairwise constraints group, and the semi-supervised clustering with label constraints. SSSE outperforms its unsupervised baseline SE and achieves the best performance among most datasets in all three groups. Specifically, for the three small datasets, SSSE achieves the best performance in all three groups, except for SSSE with label constraints on the ORL dataset, which achieves comparable ARI values but significantly higher NMI and ACC values than U-SPEC and U-SENC. For the two medium datasets, SSSE achieves the best performance on the COIL100 dataset and achieves the second-best performance on the USPS dataset. The sampling-based scalability strategy boosts the efficiency and scalability of SSSE on these datasets but inevitably sacrifices the clustering accuracy. For the two large datasets, SE and most semi-supervised clustering methods fail to give the clustering results owing to high time or space complexity. SSSE outperforms scalable unsupervised clustering methods and Seeded KMeans on these two datasets. In addition, we find that for datasets ORL and USPS, the unsupervised methods U-SPEC and U-SENC achieve promising performance and outperform many semi-supervised clustering methods. However, the performance of most semi-supervised methods, including

[1]https://github.com/SELGroup/SSSE

TABLE III
PERFORMANCE (DP) OF METHODS FOR SEMI-SUPERVISED HIERARCHICAL CLUSTERING ON CLUSTERING DATASETS. **BOLD**: THE BEST PERFORMANCE .

Method(%)	Yale	Wine	ORL	Australian	Isolet	COIL100	USPS	MNIST	EMNIST
SpecWRSC	**47.25**$_{\pm 1.33}$	87.48$_{\pm 0.79}$	OOM	65.61$_{\pm 1.24}$	OOM	OOM	OOM	OOM	OOM
SE	40.03$_{\pm 1.28}$	86.23$_{\pm 0.43}$	58.74$_{\pm 2.00}$	55.17$_{\pm 0.70}$	41.39$_{\pm 0.67}$	**67.41**$_{\pm 1.51}$	46.73$_{\pm 3.91}$	N/A	N/A
SE$_{scalable}$	39.63$_{\pm 1.43}$	86.61$_{\pm 0.65}$	56.84$_{\pm 1.76}$	55.08$_{\pm 1.00}$	40.70$_{\pm 1.14}$	61.71$_{\pm 1.97}$	53.85$_{\pm 2.34}$	28.59$_{\pm 3.52}$	07.31$_{\pm 0.54}$
COBRA	24.36$_{\pm 1.80}$	75.67$_{\pm 11.78}$	24.30$_{\pm 1.88}$	63.85$_{\pm 4.25}$	**49.98**$_{\pm 3.70}$	17.26$_{\pm 1.44}$	57.63$_{\pm 5.79}$	**40.89**$_{\pm 2.79}$	N/A
Semi-Multicons	32.70$_{\pm 2.91}$	89.02$_{\pm 7.61}$	23.55$_{\pm 1.66}$	69.71$_{\pm 2.41}$	N/A	N/A	N/A	N/A	N/A
SSSE (Ours)	44.83$_{\pm 1.85}$	**92.15**$_{\pm 1.00}$	**62.98**$_{\pm 1.78}$	**71.87**$_{\pm 1.61}$	48.17$_{\pm 1.71}$	62.08$_{\pm 1.64}$	**61.05**$_{\pm 3.06}$	33.39$_{\pm 3.12}$	**07.56**$_{\pm 0.58}$

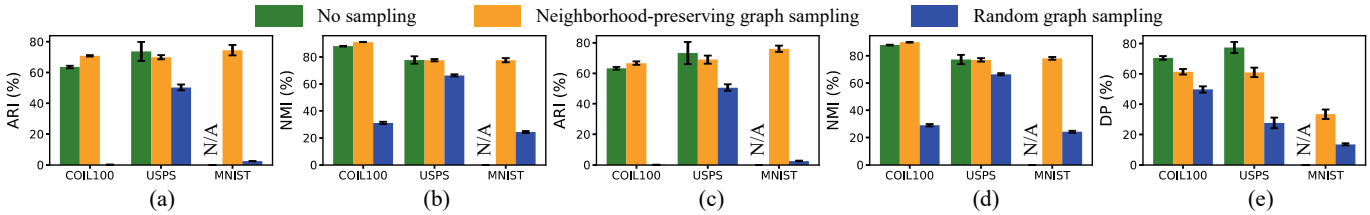

Fig. 4. Performance of SSSE with no sampling, our neighborhood-preserving graph sampling strategy and random sampling strategy. (a) and (b): ARI and NMI values of SSSE results for semi-supervised partitioning clustering with pairwise constraints. (c) and (d): ARI and NMI values for semi-supervised partitioning clustering with label constraints. (e): DP values for semi-supervised hierarchical clustering with pairwise constraints.

SSSE, increases along with more constraints. Semi-supervised methods potentially outperform unsupervised methods when more constraints are given. We also find that the unsupervised version SE, which minimizes structural entropy only without prior information, achieves quite high performance, which contributes to the superior performance of SSSE. Comparing SSSE with different constraints, SSSE with label constraints underperforms compared to SSSE with pairwise constraints in most cases. Furthermore, SSSE with label constraints only outperforms its baseline SE by a small margin in some cases. This suggests that the constraint conversion and relation graph sparsification retain only a portion of the prior information. In all, SSSE effectively integrates prior information of pairwise and label constraints, and efficiently outputs clustering results with high accuracy for datasets at different scales.

B. Semi-Supervised Hierarchical Clustering

In this part, we evaluate the performance of SSSE on semi-supervised hierarchical clustering. We adopt a widely used hierarchical clustering metric Dendrogram Purity (DP) [41] for performance evaluation. DP is a holistic measure of a cluster tree, which is defined as the average purity score of ancestors of all leaf pairs with the same ground truth labels. We compare SSSE with three unsupervised hierarchical clustering methods (SpecWRSC [42], SE [24], and SE$_{scalable}$) and two semi-supervised hierarchical clustering methods (COBRA [17] and Semi-Multicons [43]).

Implementation Details. For a given dataset $\mathcal{X}$ with n data points, we construct a p-nearest neighbor graph using cosine similarity and empirically set $p = 7$. The number of sampling seeds is empirically set to be $n_{seed} = 10$ and the sampling size is set to be $n_s = 1000$. We generate constraints using the ground truth class labels from each dataset. We generate $0.2n$ must-link constraints and $0.2n$ cannot-link constraints randomly for semi-supervised clustering methods. We simply set $\phi = 1$. The experiments are conducted 10 times on a server with two Intel(R) 2.30 GHz CPUs and 500 GB RAM.

Experimental Results. In Table III, we show the performance of methods for hierarchical clustering on 9 clustering datasets. SSSE achieves the best or second-best performance on all datasets. In particular, SpecWRSC and Semi-Multicons are not able to work on most datasets since they have high space complexity and time complexity, respectively. COBRA achieves comparable performance to SSSE on two datasets. However, it still can not tackle the largest dataset EMNIST, making it inferior to SSSE at scalability. COBRA and Semi-Multicons underperform compared to SSSE on most datasets because they fail to fully capture the hierarchical structure of the datasets. Specifically, COBRA builds cluster trees based on super-instances via the KMeans algorithm, where each super-instance contains several data points. The hierarchical structures of data points within super-instances are under-captured. Semi-Multicons, on the other hand, generates cluster trees at a given height and cannot accommodate a large height value. The tree node usually contains many children nodes, making the hierarchical structures of children nodes under-captured. In all, SSSE achieves high accuracy and scalability at semi-supervised hierarchical clustering.

C. Further Analysis

Ablation study on sampling strategy. The graph sampling strategy improves the scalability of SSSE but inevitably sacrifices the performance. We propose the neighborhood-preserving graph sampling strategy to preserve the intrinsic neighborhood structure of the data graph. To verify the effectiveness of this strategy, we evaluate the performance of SSSE without sampling, with our sampling strategy, and with the random sampling strategy, as shown in Fig. 4. We conduct experiments on three datasets for three tasks: semi-supervised partitioning clustering with pairwise constraints, semi-supervised partitioning clustering with label constraints, and semi-supervised hierarchical clustering with pairwise constraints. We observe that our proposed sampling strategy greatly improves the performance of SSSE on all datasets for

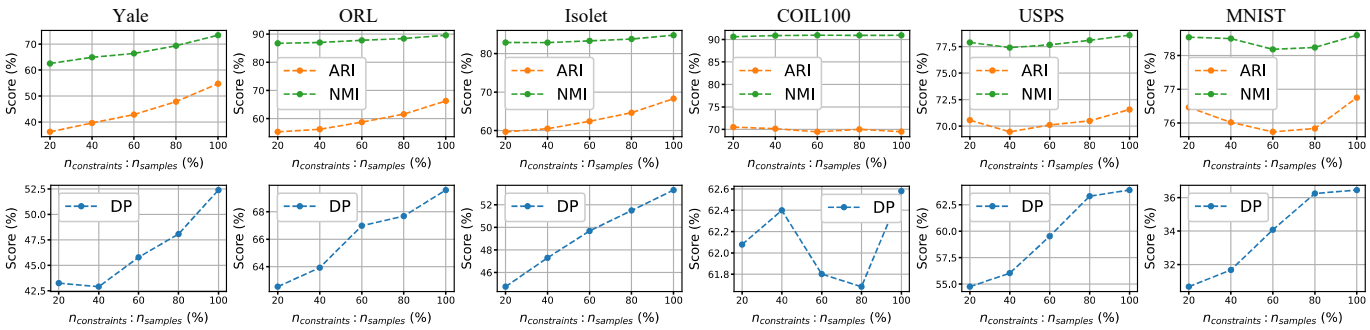

Fig. 5. Performance of SSSE for semi-supervised clustering with different pairwise constraint amounts. First row: semi-supervised partitioning clustering. Second row: semi-supervised hierarchical clustering.

TABLE IV
TIME COSTS(S) OF METHODS FOR PARTITIONING CLUSTERING ON SEVEN CLUSTERING DATASETS. **BOLD**: THE BEST PERFORMANCE.

Cat.	Method	Yale	ORL	Isolet	COIL100	USPS	MNIST	EMNIST
Unsup.	GBSC	**0.08**$_{\pm 0.02}$	**0.19**$_{\pm 0.00}$	2.32$_{\pm 0.06}$	46.81$_{\pm 0.13}$	73.26$_{\pm 0.20}$	4547.45$_{\pm 290.70}$	OOM
	U-SPEC	0.21$_{\pm 0.25}$	0.30$_{\pm 0.03}$	**0.89**$_{\pm 0.63}$	**3.56**$_{\pm 0.27}$	**1.75**$_{\pm 0.09}$	**6.65**$_{\pm 0.44}$	**24.30**$_{\pm 3.02}$
	U-SENC	1.93$_{\pm 0.49}$	4.59$_{\pm 0.19}$	14.11$_{\pm 0.43}$	49.56$_{\pm 3.61}$	29.25$_{\pm 0.15}$	108.84$_{\pm 2.44}$	332.27$_{\pm 144.36}$
	SE	1.73$_{\pm 4.21}$	0.52$_{\pm 0.06}$	4.67$_{\pm 4.14}$	32.10$_{\pm 2.50}$	569.47$_{\pm 199.18}$	N/A	N/A
Pairwise	PCPSNMF	0.46$_{\pm 0.13}$	4.83$_{\pm 0.95}$	119.41$_{\pm 39.12}$	OOM	OOM	OOM	OOM
	OneStepPCP	1.25$_{\pm 0.67}$	4.71$_{\pm 0.06}$	61.33$_{1.23}$	4266.46$_{\pm 328.83}$	9997.89$_{\pm 645.90}$	N/A	N/A
	CMS	9.39$_{\pm 1.70}$	47.15$_{\pm 1.17}$	448.56$_{\pm 8,86}$	16924.96$_{\pm 255.02}$	10644.05$_{\pm 366.79}$	N/A	N/A
	SC-MPI	**0.16**$_{\pm 0.36}$	**0.12**$_{\pm 0.01}$	**1.37**$_{\pm 0.52}$	37.82$_{\pm 10.08}$	106.31$_{\pm 1.38}$	OOM	OOM
	SSSE (Ours)	1.67$_{\pm 4.09}$	0.63$_{\pm 0.05}$	6.54$_{\pm 4.24}$	**31.36**$_{\pm 5.40}$	**39.31**$_{\pm 1.55}$	**209.09**$_{\pm 32.00}$	**399.03**$_{\pm 30.32}$
Label	Seeded KMeans	**0.04**$_{\pm 0.03}$	**0.05**$_{\pm 0.01}$	**0.05**$_{\pm 0.05}$	**1.39**$_{\pm 0.33}$	**0.26**$_{\pm 0.13}$	**5.87**$_{\pm 0.79}$	**12.47**$_{\pm 3.93}$
	S4NMF	57.84$_{\pm 10.31}$	75.36$_{\pm 1.58}$	172.26$_{\pm 0.65}$	4119.54$_{\pm 2.52}$	5184.86$_{\pm 0.44}$	N/A	N/A
	SC-MPI	0.09$_{\pm 0.02}$	0.09$_{\pm 0.00}$	0.32$_{\pm 0.03}$	8.38$_{\pm 0.19}$	8.49$_{\pm 0.36}$	OOM	OOM
	SSSE (Ours)	2.82$_{\pm 7.30}$	0.67$_{\pm 0.10}$	6.06$_{\pm 4.24}$	23.97$_{\pm 4.25}$	21.72$_{\pm 1.47}$	216.95$_{\pm 6.32}$	402.13$_{\pm 26.74}$

TABLE V
TIME COSTS(S) OF METHODS FOR HIERARCHICAL CLUSTERING ON NINE CLUSTERING DATASETS. **BOLD**: THE BEST PERFORMANCE.

Method	Yale	Wine	ORL	Australian	Isolet	COIL100	USPS	MNIST	EMNIST
SpecWRSC	3.55$_{\pm 4.13}$	1.93$_{\pm 4.29}$	OOM	5.66$_{\pm 4.19}$	OOM	OOM	OOM	OOM	OOM
SE	1.56$_{\pm 4.19}$	1.70$_{\pm 4.15}$	0.34$_{\pm 0.05}$	10.52$_{\pm 0.35}$	5.69$_{\pm 0.29}$	32.80$_{\pm 0.52}$	286.7$_{\pm 10.0}$	N/A	N/A
$SE_{scalable}$	1.71$_{\pm 4.20}$	1.72$_{\pm 4.25}$	0.67$_{\pm 0.10}$	10.40$_{\pm 0.19}$	**5.39**$_{\pm 0.17}$	10.57$_{\pm 0.72}$	19.69$_{\pm 2.30}$	359.47$_{\pm 27.58}$	555.36$_{\pm 37.27}$
COBRA	1.98$_{\pm 0.45}$	**0.19**$_{\pm 0.00}$	3.80$_{\pm 0.70}$	**1.31**$_{\pm 0.11}$	7.91$_{\pm 0.46}$	309.0$_{\pm 37.4}$	473.2$_{\pm 60.3}$	5104$_{\pm 2111}$	N/A
Semi-Multicons	876$_{\pm 41}$	138.9$_{\pm 10.8}$	5404$_{\pm 1275}$	2124$_{\pm 106}$	N/A	N/A	N/A	N/A	N/A
SSSE (Ours)	**1.54**$_{\pm 4.18}$	1.75$_{\pm 4.26}$	**0.33**$_{\pm 0.05}$	12.60$_{\pm 4.22}$	6.73$_{\pm 4.54}$	**5.27**$_{\pm 15.00}$	**8.22**$_{\pm 1.11}$	**270.40**$_{\pm 67.43}$	**555.01**$_{\pm 43.67}$

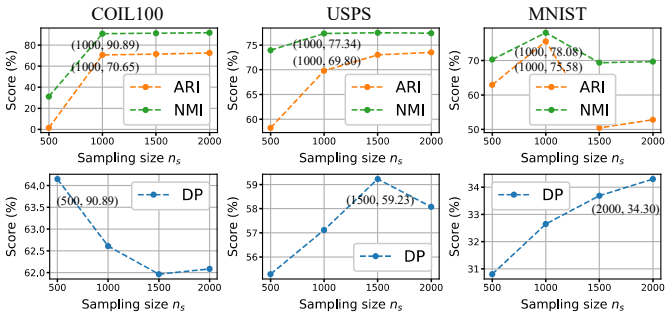

Fig. 6. Performance of SSSE with pairwise constraints at different sampling sizes. First row: semi-supervised partitioning clustering. Second row: semi-supervised hierarchical clustering.

all tasks compared to the random sampling strategy. Specifically, ARI values are nearly 0 on the COIL100 and MNIST datasets with the random sampling strategy. We also find that although our proposed sampling strategy demonstrates a performance largely comparable to the no sampling version of SSSE, i.e., SSE in our conference paper [23], it does exhibit a slight decrease in performance, particularly in semi-supervised hierarchical clustering. In all, our proposed neighborhood-preserving graph sampling strategy effectively preserves the intrinsic neighborhood structure of the data graph and greatly boosts the performance of SSSE compared to the random sampling strategy.

Constraints number. The number of constraints is known to have a great impact on the performance of semi-supervised clustering methods. We evaluate the performance of SSSE with different amounts of constraints for three tasks, which are presented in Fig. 5. The performance of SSSE is higher with more constraints on most datasets, indicating that constraints are properly integrated into SSSE. Specifically, for three small datasets, the performance of SSSE witnesses a clear increase with larger constraint amounts on all three tasks. For the other three datasets, the graph sampling step weakens the effect of constraints since only a small proportion of edges in the

relation graph G' are selected in the first round of sampling. However, the performance of SSSE still witnesses a trend of increase with a larger amount of constraints, especially for semi-supervised hierarchical clustering on the USPS and MNIST datasets.

Sampling size. We analyze the effect of sampling size n_s on the performance of SSSE for the tasks of semi-supervised partitioning and hierarchical clustering, which are shown in Fig. 6. With some exceptions, the performance of SSSE is usually higher when the sampling size n_s is larger, which indicates that larger n_s helps preserve more graph structure in sampled subgraphs. However, a large n_s leads to large time and space complexity, which decreases the scalability of SSSE. We find that $n_s = 1000$ is a good choice for SSSE.

Efficiency. We show the time costs of methods for semi-supervised partitioning and hierarchical clustering in Table IV and Table V, respectively. For the task of partitioning clustering, the time cost of SSSE is comparable to three scalable unsupervised clustering methods, and even lower than GBSC on medium and large datasets. Specifically, in the group of methods with pairwise constraints, the time cost of SSSE is lowest on medium and large datasets. In the group of methods with label constraints, SSSE along with Seeded KMeans are the only two methods able to tackle the largest dataset EMNIST. For the task of hierarchical clustering, the time cost of SSSE is the lowest on most datasets. In all, SSSE is a scalable method with high efficiency, especially for medium and large datasets.

V. Application: Semi-Supervised Clustering of Biological Data at Different Scales

We investigate the functionality of SSSE for semi-supervised clustering of biological data at different scales. We evaluate the performance of SSSE on single-cell RNA-seq data, the most used data for cell type determination. Semi-supervised clustering algorithms provide the possibility for utilizing additional biological prior information such as marker genes for higher cell type identification performance [5]. We conduct experiments on 7 datasets divided into three groups: small datasets (10X PBMC, Mouse bladder, Human kidney), medium datasets (Human liver, CITE PBMC, Baron(human)), and a large dataset (Karagiannis), containing cells ranging from 2100 to 72914. Among them, three small datasets are derived from Tian *et al.* [5], which were preprocessed by the authors to contain 2100 randomly sampled cells. We retain the data with the top 2000 genes for all datasets to eliminate the interference of less expressed irrelevant genes. We compare SSSE with partitioning clustering baselines, including several scalable unsupervised clustering methods (GBSC [36], U-SPEC [37], U-SENC [37], SE$_{scalable}$), several semi-supervised clustering methods with pairwise constraints (PCPSNMF [38], OneStepPCP [39], CMS [9]), several semi-supervised clustering methods with label constraints (Seeded-KMeans [1], S4NMF [40], and one semi-supervised clustering method with both pairwise and label constraints (SC-MPI [3]). Unlike the experiments in Sec. IV-A, we adopt the cosine similarity for RNA-seq datasets since the features of these datasets are sparse. All other parameters and experimental settings are the same as in Sec. IV-A.

Performance evaluation. In Table VI, we show the performance of methods for partitioning clustering on single-cell RNA-seq datasets. The experimental results are compared separately in three groups. SSSE achieves the best performance on most datasets among methods in all three groups, demonstrating the functionality of SSSE for semi-supervised clustering of RNA-seq data. Meanwhile, SSSE outperforms the unsupervised method SE$_{scalable}$ on all datasets, indicating that constraints boost clustering accuracy by being effectively integrated into SSSE. For three small datasets, we observe that SSSE achieves the best performance in all three groups. For the other four datasets, the performance of SSSE sacrifices for scalability and efficiency through graph sampling. However, SSSE still achieves the best performance on three datasets and achieves comparable performance on the Baron(human) dataset. Moreover, most semi-supervised clustering methods are unable to tackle the large dataset Karagiannis due to high time or space complexity. In all, SSSE effectively performs semi-supervised partitioning clustering on single-cell RNA-seq datasets with high accuracy.

To further analyze the performance of SSSE on biological data, we visualize six RNA-seq datasets along with the clustering results, as shown in Fig. 7. For each dataset, We build a data graph G used in SSSE and utilize the spring embedder by the Fruchterman-Reginold force-directed algorithm [44] to calculate the positions of vertices in G. The colors of points represent the ground truth labels (first row in Fig. 7), predicted labels by SSSE with pairwise constraints (second row), and predicted labels by SSSE with label constraints (last row). The spring embedder separates the ground truth cell type for most datasets, and SSSE rightly clusters most of the cell types. However, some cell types stick to each other, resulting in suboptimal separation when subjected to SSSE. In addition, some cell types have very small numbers of cells and are interspersed among other cell types, making them clustered as a part of other cell types. For the dataset Baron(human), the constructed data graph G fails to separate many of the cell types, leading to poor performance of SSSE on this dataset.

VI. Related Works

A. Semi-Supervised Clustering

Most of the existing semi-supervised clustering methods leverage prior information in two approaches: (1) They strive for a clustering outcome that aligns with given side information by integrating a regularization term into the initial clustering objective. (2) They seek to derive more informative similarity (or distance) by learning an adjusted similarity measure from the given side information [15], [16]. In this work, our focus is the first approach.

Prior information can take different forms of constraints, among them pairwise constraints are the most widely used. Wagstaff *et al.* [45] introduced the concept of constrained clustering using pairwise constraints in 2000 and proposed a centroid-based semi-supervised clustering algorithm COP-KMeans [12] in the next year. Hereafter, numerous semi-supervised clustering methods based on various conventional

TABLE VI
PERFORMANCE OF METHODS FOR SEMI-SUPERVISED CLUSTERING ON SINGLE-CELL RNA-SEQ DATASETS. **BOLD**: THE BEST PERFORMANCE .

Metric	Cat.	Method (%)	10X PBMC	Mou. bladder	Hum. kidney	Hum. liver	CITE PBMC	Baron	Karagiannis
ARI	Unsup.	GBSC	33.49±0.00	21.38±0.29	04.30±0.82	00.06±0.06	13.58±10.7	00.56±0.12	OOM
		U-SPEC	22.31±8.36	39.60±3.78	00.25±0.04	00.26±0.10	00.77±0.14	42.18±2.89	33.19±2.60
		U-SENC	29.25±2.20	39.48±2.30	01.34±1.16	34.28±16.6	04.70±0.03	38.20±2.41	31.41±0.97
		$SE_{scalable}$	64.45±0.05	60.30±0.05	52.93±0.14	68.01±14.6	59.73±2.27	47.02±0.71	48.73±4.67
	Pairwise	PCPSNMF	20.52±9.61	14.52±3.22	16.28±4.47	OOM	OOM	OOM	OOM
		OneStepPCP	43.12±0.06	44.18±4.19	41.48±2.60	56.51±9.78	49.12±3.97	**60.79**±3.76	N/A
		CMS	09.38±4.33	08.28±0.52	07.92±4.51	N/A	N/A	N/A	N/A
		SC-MPI	27.75±13.1	18.79±3.18	16.54±2.35	11.52±4.58	19.84±3.51	36.27±8.52	OOM
		SSSE (Ours)	**68.42**±3.68	**66.88**±1.70	**68.63**±7.81	**80.89**±8.36	**62.03**±0.97	47.80±3.79	**56.03**±1.58
	Label	Seeded KMeans	67.03±1.51	38.55±4.61	17.35±0.36	18.12±5.67	50.19±0.38	18.60±0.36	38.35±0.00
		S4NMF	18.39±1.47	26.71±0.90	24.64±0.99	OOM	OOM	OOM	OOM
		SC-MPI	54.28±6.46	43.13±10.7	43.71±24.3	**93.48**±1.98	61.04±2.36	**77.15**±7.28	OOM
		SSSE (Ours)	**67.46**±0.80	**61.76**±0.79	**61.61**±4.18	73.13±12.7	**61.75**±1.00	50.07±6.76	**55.49**±1.41
NMI	Unsup.	GBSC	50.18±3.51	42.76±0.14	15.05±3.18	02.13±0.05	28.05±12.6	02.98±0.29	OOM
		U-SPEC	40.28±12.4	62.37±1.56	02.40±0.22	02.16±0.37	04.41±0.71	64.50±0.58	60.14±1.71
		U-SENC	47.39±2.26	64.67±1.00	07.52±3.99	58.13±8.18	19.53±0.10	65.02±1.42	61.15±0.55
		$SE_{scalable}$	75.33±0.03	74.46±0.04	72.02±0.01	74.05±3.78	67.38±1.45	58.79±3.46	55.26±1.83
	Pairwise	PCPSNMF	34.72±11.6	39.61±4.39	30.40±4.12	OOM	OOM	OOM	OOM
		OneStepPCP	58.26±0.21	64.48±1.85	55.68±1.44	68.98±1.59	68.49±1.91	**75.95**±1.45	N/A
		CMS	22.82±7.43	10.56±3.69	12.26±6.74	N/A	N/A	N/A	N/A
		SC-MPI	39.62±13.4	40.70±2.76	30.82±2.36	25.92±3.04	38.93±3.37	50.56±5.27	OOM
		SSSE (Ours)	**76.26**±0.85	**76.41**±0.51	**77.80**±3.05	**76.27**±2.56	**69.95**±1.49	59.01±1.68	**55.97**±1.17
	Label	Seeded KMeans	70.79±0.93	62.24±3.10	39.66±0.63	44.13±3.46	66.07±0.11	37.78±0.80	**63.19**±0.00
		S4NMF	28.32±1.43	44.08±0.99	39.44±1.46	OOM	OOM	OOM	OOM
		SC-MPI	65.32±2.17	58.08±0.56	53.40±13.9	**86.33**±1.06	68.56±1.59	**75.98**±3.09	OOM
		SSSE (Ours)	**76.33**±0.46	**75.11**±0.28	**75.87**±1.62	74.83±2.54	**70.47**±1.55	60.07±2.96	55.77±1.17

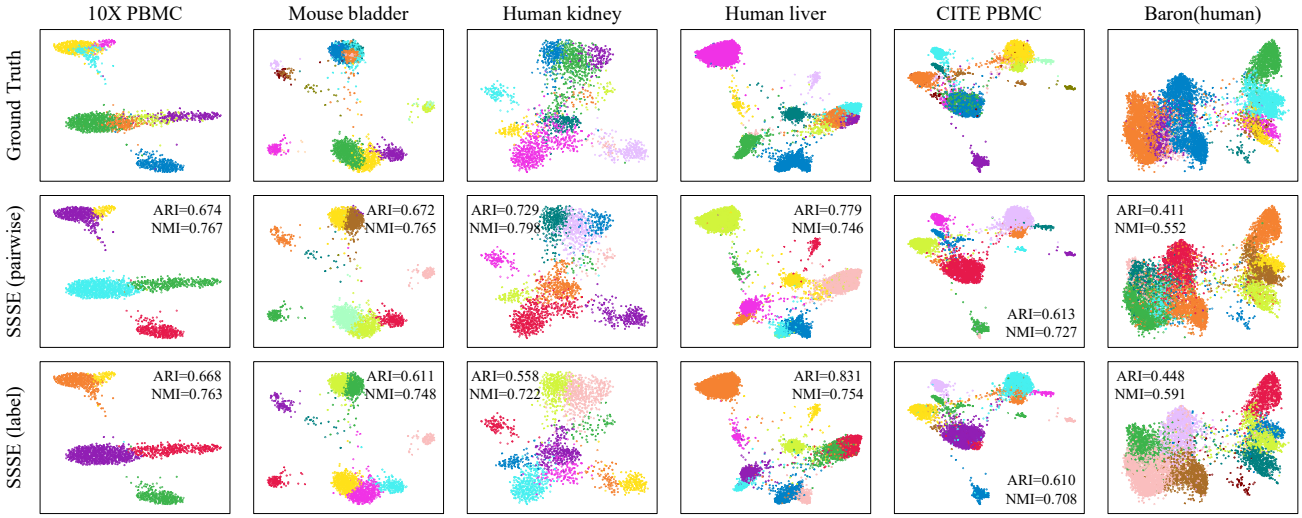

Fig. 7. Visualization of single-cell RNA-seq datasets.

clustering have been proposed, such as KMeans-based [46], Spectral clustering-based [10], density-based [9], and NMF-based [38] methods. More recently, Jia *et al.* [39] pointed out that the two-step manner of most constrained spectral clustering algorithms based on pairwise constraint propagation (PCP) leads to sub-optimal clustering output, and proposed a joint PCP model by unifying these two steps of propagation matrix learning and affinity matrix learning. Schier *et al.* [9] proposed a semi-supervised mean shift clustering algorithm using cannot-link constraints, this algorithm generates individual distributions of the sampling points per cluster through a density-based integration.

Prior information can also take the form of label constraints. Semi-supervised clustering algorithms based on label constraints originate by Basu *et al.* from Seeded-KMeans [1] and go through a long-term development in the following two decades [7], [47] In the recent past, Jiang *et al.* [7] presented a "Compact-cluster" assumption, i.e., without low-density separation inside a cluster, for semi-supervised clustering, and proposed the CSSC framework via cluster-splitting technique. Liu *et al.* [48] proposed a semi-supervised NMF clustering method LpNMF which uses data's intrinsic geometric structure in the regularization term of its objective.

There exist some other types of constraints, such as triplet

constraints [8], size constraints [14], and existential cluster constraints [49]. In the big data era, many datasets have different types, which calls for semi-supervised clustering methods to be able to tackle more than one type of prior information. Only a few methods [3], [50] consider multiple types of constraints at the same time, among them our previous work SSE [23] uniforms pairwise constraints and label constraints into one relation graph.

B. Scalable Clustering

Scalability is important for clustering in the big data era, especially for graph-based clustering methods. There exists a long list of scalable clustering methods [17], [18], [20], [51], [52], most of which take specific scalability strategies designed for different clustering methods. A widely used scalability strategy is clustering size reduction [17], [18], [51]. For example, Monath *et al.* [51] propose a nearest neighbor-based subcluster component algorithm to form sub-clusters and reduce clustering size, achieving scalable agglomerative hierarchical clustering scaling to billions of data points. Another strategy is anchor graph generation [20], [52], where a small number of anchors are selected to construct a data-to-anchor graph. One example is the landmark-based sparse representation (LSR) proposed by Cai *et al.* [52] for large-scale spectral clustering, where the original data points can be represented by sparse linear combinations of KMeans chosen landmarks.

The scalability of semi-supervised clustering methods is less discussed [17], [21]. Van *et al.* [17] proposed COBRA for fast active clustering with pairwise constraints. It over-clusters the data via KMeans to reduce the data size, and sequentially merge clusters by querying pairwise constraints. Gao *et al.* [21] proposed an efficient local search algorithm FastCCP for constrained clustering. It first merges instances connected by must-link constraints, and then performs local search for clustering based on cannot-link constraints. However, all of them fail to handle super-large datasets.

VII. Conclusion

This paper proposes a novel scalable and general semi-supervised clustering method via structural entropy, namely SSSE. It enables scalability on data size via graph sampling and generalizability on constraint types by giving a uniform formulation of different constraints. Specifically, we design objectives based on structural entropy along with a regularization term to integrate constraints. We formulate pairwise and label constraints as a uniform view and store them in a relation graph utilized in the regularization term. We apply a sampling-based scalability strategy to achieve efficient and scalable clustering and propose a neighborhood-preserving sampling strategy to alleviate the impedance of sampling on clustering accuracy. Furthermore, we design objectives and optimization algorithms for semi-supervised partitioning and hierarchical clustering based on 2-D and high-D structural entropy, respectively. We conduct extensive experiments on real-world clustering datasets at different scales, verifying the effectiveness and efficiency of SSSE. We also apply SSSE to single-cell RNA-seq datasets for cell clustering, demonstrating its functionality in biological data analysis.

Acknowledgments

This work is supported by the National Key R&D Program of China through grant 2022YFB3104703, NSFC through grants 61932002, 62322202, and 62432006, Local Science and Technology Development Fund of Hebei Province Guided by the Central Government of China through grant 246Z0102G, Guangdong Basic and Applied Basic Research Foundation through grant 2023B1515120020, CCF-DiDi GAIA Collaborative Research Funds for Young Scholars, NSF under grants III-2106758, and POSE-2346158.

References

[1] S. Basu, A. Banerjee, and R. J. Mooney, "Semi-supervised clustering by seeding," in *Proceedings of the ICML*, 2002, pp. 27–34.
[2] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM CSUR*, vol. 31, no. 3, pp. 264–323, 1999.
[3] L. Bai, J. Liang, and F. Cao, "Semi-supervised clustering with constraints of different types from multiple information sources," *IEEE TPAMI*, vol. 43, no. 9, pp. 3247–3258, 2020.
[4] S. E. Chew and N. D. Cahill, "Semi-supervised normalized cuts for image segmentation," in *Proceedings of the IEEE ICCV*, 2015, pp. 1716–1723.
[5] T. Tian, J. Zhang, X. Lin, Z. Wei, and H. Hakonarson, "Model-based deep embedding for constrained clustering analysis of single cell rna-seq data," *Nat. Commun.*, vol. 12, no. 1, p. 1873, 2021.
[6] C. Dai, J. Wu, J. J. Monaghan, G. Li, H. Peng, S. I. Becker, and D. McAlpine, "Semi-supervised eeg clustering with multiple constraints," *IEEE TKDE*, vol. 35, no. 8, pp. 8529–8544, 2023.
[7] Z. Jiang, Y. Zhan, Q. Mao, and Y. Du, "Semi-supervised clustering under a "compact-cluster" assumption," *IEEE TKDE*, vol. 35, no. 5, pp. 5244–5256, 2022.
[8] V. Chatziafratis, R. Niazadeh, and M. Charikar, "Hierarchical clustering with structural constraints," in *Proceedings of the ICML*, 2018, pp. 774–783.
[9] M. Schier, C. Reinders, and B. Rosenhahn, "Constrained mean shift clustering," in *Proceedings of the SDM*, 2022, pp. 235–243.
[10] F. Nie, H. Zhang, R. Wang, and X. Li, "Semi-supervised clustering via pairwise constrained optimal graph," in *Proceedings of the IJCAI*, 2021, pp. 3160–3166.
[11] J. Yin, S. Peng, Z. Yang, B. Chen, and Z. Lin, "Hypergraph based semi-supervised symmetric nonnegative matrix factorization for image clustering," *Pattern Recognition*, vol. 137, p. 109274, 2023.
[12] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl *et al.*, "Constrained k-means clustering with background knowledge," in *Proceedings of the ICML*, vol. 1, 2001, pp. 577–584.
[13] L. Zheng and T. Li, "Semi-supervised hierarchical clustering," in *Proceedings of the IEEE ICDM*, 2011, pp. 982–991.
[14] S. Zhu, D. Wang, and T. Li, "Data clustering with size constraints," *Knowledge-Based Systems*, vol. 23, no. 8, pp. 883–889, 2010.
[15] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proceedings of the ICML*, 2004, pp. 81–88.
[16] W. Kalintha, S. Ono, M. Numao, and K.-i. Fukui, "Kernelized evolutionary distance metric learning for semi-supervised clustering," in *Proceedings of the AAAI*, 2017, pp. 4945–4946.
[17] T. Van Craenendonck, S. Dumančic, and H. Blockeel, "Cobra: a fast and simple method for active clustering with pairwise constraints," in *Proceedings of the IJCAI*, 2017, pp. 2871–2877.
[18] H. Shinnou and M. Sasaki, "Spectral clustering for a large data set by reducing the similarity matrix size." in *Proceedings of the LREC*, 2008, pp. 201–204.
[19] C. Liu, F. Nie, R. Wang, and X. Li, "Scalable fuzzy clustering with anchor graph," *IEEE TKDE*, vol. 35, no. 8, pp. 8503–8514, 2023.
[20] F. Nie, J. Xue, W. Yu, and X. Li, "Fast clustering with anchor guidance," *IEEE TPAMI*, no. 01, pp. 1–15, 2023.
[21] J. Gao, X. Tao, and S. Cai, "Towards more efficient local search algorithms for constrained clustering," *Information Sciences*, vol. 621, pp. 287–307, 2023.
[22] Y. Cao, H. Peng, Z. Yu, and P. S. Yu, "Hierarchical and incremental structural entropy minimization for unsupervised social event detection," in *Proceedings of the AAAI*, 2024, pp. 1–9.

[23] G. Zeng, H. Peng, A. Li, Z. Liu, R. Yang, C. Liu, and L. He, "Semi-supervised clustering via structural entropy with different constraints," in *Proceedings of the SDM*, 2024, pp. 1–10.
[24] A. Li and Y. Pan, "Structural information and dynamical complexity of networks," *IEEE TIT*, vol. 62, no. 6, pp. 3290–3339, 2016.
[25] L. Chen and S. C. Li, "Incorporating cell hierarchy to decipher the functional diversity of single cells," *Nucleic acids research*, vol. 51, no. 2, pp. e9–e9, 2023.
[26] D. Zou, H. Peng, X. Huang, R. Yang, J. Li, J. Wu, C. Liu, and P. S. Yu, "Se-gsl: A general and effective graph structure learning framework through structural entropy optimization," in *Proceedings of the ACM Web Conference*, 2023, pp. 499–510.
[27] X. Zeng, H. Peng, and A. Li, "Adversarial socialbots modeling based on structural information principles," in *Proceedings of the AAAI*, vol. 38, no. 1, 2024, pp. 392–400.
[28] Y. Pan, F. Zheng, and B. Fan, "An information-theoretic perspective of hierarchical clustering," *arXiv preprint arXiv:2108.06036*, 2021.
[29] L. Sun, Z. Huang, H. Peng, Y. Wang, C. Liu, and S. Y. Philip, "Lsenet: Lorentz structural entropy neural network for deep graph clustering," in *Proceedings of the ICML*, 2024, pp. 1–10.
[30] J.-P. Vert, K. Tsuda, and B. Schölkopf, "A primer on kernel methods," *Kernel methods in computational biology*, vol. 47, pp. 35–70, 2004.
[31] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proceedings of the ACM Web Conference*, 2011, pp. 577–586.
[32] S. Shen, W. Li, Z. Zhu, J. Zhou, and J. Lu, "Star-fc: Structure-aware face clustering on ultra-large-scale graphs," *IEEE TPAMI*, vol. 45, no. 11, pp. 14 005–14 019, 2023.
[33] C. Deng. "codes and datasets for feature learning," 2024. [Online]. Available: http://www.cad.zju.edu.cn/home/dengcai/Data/data.html
[34] K. Markelle, L. Rachel, and N. Kolby. The uci machine learning repository. [Online]. Available: https://archive.ics.uci.edu
[35] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *Proceedings of the IJCNN*. IEEE, 2017, pp. 2921–2926.
[36] J. Xie, W. Kong, S. Xia, G. Wang, and X. Gao, "An efficient spectral clustering algorithm based on granular-ball," *IEEE TKDE*, vol. 35, no. 9, pp. 9743–9753, 2023.
[37] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, and C.-K. Kwoh, "Ultra-scalable spectral clustering and ensemble clustering," *IEEE TKDE*, vol. 32, no. 6, pp. 1212–1226, 2019.
[38] W. Wu, Y. Jia, S. Kwong, and J. Hou, "Pairwise constraint propagation-induced symmetric nonnegative matrix factorization," *IEEE TNNLS*, vol. 29, no. 12, pp. 6348–6361, 2018.
[39] Y. Jia, W. Wu, R. Wang, J. Hou, and S. Kwong, "Joint optimization for pairwise constraint propagation," *IEEE TNNLS*, vol. 32, no. 7, pp. 3168–3180, 2020.
[40] J. Chavoshinejad, S. A. Seyedi, F. A. Tab, and N. Salahian, "Self-supervised semi-supervised nonnegative matrix factorization for data clustering," *Pattern Recognition*, vol. 137, p. 109282, 2023.
[41] N. Yadav, A. Kobren, N. Monath, and A. Mccallum, "Supervised hierarchical clustering with exponential linkage," in *Proceedings of the ICML*, 2019, pp. 6973–6983.
[42] S. Laenen, B. A. Manghiuc, and H. Sun, "Nearly-optimal hierarchical clustering for well-clustered graphs," in *Proceedings of the ICML*, 2023, pp. 18 207–18 249.
[43] T. Yang, N. Pasquier, and F. Precioso, "Semi-supervised consensus clustering based on closed patterns," *Knowledge-Based Systems*, vol. 235, p. 107599, 2022.
[44] S. G. Kobourov, "Spring embedders and force directed graph drawing algorithms," *arXiv preprint arXiv:1201.3011*, 2012.
[45] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *Proceedings of the ICML*, 2000, pp. 1103–1110.
[46] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of the SDM*, 2004, pp. 333–344.
[47] H. Zhang and J. Lu, "Semi-supervised fuzzy clustering: A kernel-based approach," *Knowledge-Based Systems*, vol. 22, no. 6, pp. 477–481, 2009.
[48] J. Liu, Y. Wang, J. Ma, D. Han, and Y. Huang, "Constrained nonnegative matrix factorization based on label propagation for data representation," *IEEE TAI*, vol. 5, no. 02, pp. 590–601, 2024.
[49] R. Angell, N. Monath, N. Yadav, and A. McCallum, "Interactive correlation clustering with existential cluster constraints," in *Proceedings of the ICML*, 2022, pp. 703–716.
[50] W. Xiao, Y. Yang, H. Wang, T. Li, and H. Xing, "Semi-supervised hierarchical clustering ensemble and its application," *Neurocomputing*, vol. 173, pp. 1362–1376, 2016.
[51] N. Monath, K. A. Dubey, G. Guruganesh, M. Zaheer, A. Ahmed, A. McCallum, G. Mergen, M. Najork, M. Terzihan, B. Tjanaka *et al.*, "Scalable hierarchical agglomerative clustering," in *Proceedings of the ACM SIGKDD*, 2021, pp. 1245–1255.
[52] D. Cai and X. Chen, "Large scale spectral clustering via landmark-based sparse representation," *IEEE TCYB*, vol. 45, no. 8, pp. 1669–1680, 2014.

Guangjie Zeng is currently a Ph.D. candidate in the School of Computer Science and Engineering at Beihang University. His research interests include machine learning and data mining.

Hao Peng is currently a Professor at the School of Cyber Science and Technology in Beihang University. His current research interests include machine learning, deep learning, and reinforcement learning. He is the Associate Editor of the International Journal of Machine Learning and Cybernetics (IJMLC) and Neural Networks.

Angsheng Li is a professor at the School of Computer Science, Beihang University. His research areas include Computability Theory, Computational Theory, and Information Science. His current interests focus on mathematical principles of the information world and structural information principles of machine learning and intelligent machinery.

Jia Wu is currently an Associate Professor, the Research Director for the Centre for Applied Artificial Intelligence and the Director of Higher Degree Research in the School of Computing at Macquarie University, Sydney, Australia. Prof. Wu received his Ph.D. degree in computer science from the University of Technology Sydney, Australia. His current research interests include data mining and machine learning. He is the Associate Editor of IEEE TNNLS, ACM TKDD, and Neural Networks.

Chunyang Liu is currently a researcher at Didi Chuxing Technology Co., Ltd. His research interests include machine learning and data mining.

Philip S. Yu is a Distinguished Professor and the Wexler Chair in Information Technology at the Department of Computer Science, University of Illinois Chicago. He is a Fellow of the ACM and IEEE. Dr. Yu has published more than 1,600 referred conference and journal papers cited more than 200,000 times with an H-index of 201. He has applied for more than 300 patents. Dr. Yu was the Editor-in-Chief of ACM TKDD (2011-2017) and IEEE TKDE (2001-2004).