Semi-Supervised Clustering via Structural Entropy with Different Constraints

Guangjie Zeng* Hao Peng* Angsheng Li* Zhiwei Liu† Runze Yang*
Chunyang Liu‡ Lifang He§

Abstract

Semi-supervised clustering techniques have emerged as valuable tools for leveraging prior information in the form of constraints to improve the quality of clustering outcomes. Despite the proliferation of such methods, the ability to seamlessly integrate various types of constraints remains limited. While structural entropy has proven to be a powerful clustering approach with wide-ranging applications, it has lacked a variant capable of accommodating these constraints. In this work, we present Semi-supervised clustering via Structural Entropy (SSE), a novel method that can incorporate different types of constraints from diverse sources to perform both partitioning and hierarchical clustering. Specifically, we formulate a uniform view for the commonly used pairwise and label constraints for both types of clustering. Then, we design objectives that incorporate these constraints into structural entropy and develop tailored algorithms for their optimization. We evaluate SSE on nine clustering datasets and compare it with eleven semi-supervised partitioning and hierarchical clustering methods. Experimental results demonstrate the superiority of SSE on clustering accuracy with different types of constraints. Additionally, the functionality of SSE for biological data analysis is demonstrated by cell clustering experiments conducted on four single-cell RNA-seq datasets.

Keywords: Semi-Supervised Clustering, Structural Entropy, Biological Data Analysis.

1 Introduction

Clustering is a key technique in machine learning that aims to group instances according to their similarity [8]. Yet, unsupervised clustering alone often fails to provide the desired level of accuracy and may not meet the diverse requirements of various users. In contrast, semi-supervised clustering harnesses the power of prior information in the form of constraints, significantly boosting clustering accuracy and aligning more effectively with user preferences [18].

Numerous semi-supervised clustering methods based on different classical unsupervised clustering methods have been proposed in recent years. The challenges of semi-supervised clustering are 1) to design an objective function integrating constraints into clustering methods and 2) to effectively and efficiently optimize the objective. The most widely-used way to utilize the prior information is to add a regularization on the original clustering objective [12, 18]. Alternatively, some methods propagate this information to augment the dataset itself [14, 16]. The provided prior information can manifest in various constraint forms, such as pairwise constraints [24], and label constraints [17], and triplet constraints [31]. Many existing semi-supervised clustering methods are tailored to handle a single type of constraint. Yet, it is common for prior information to come in diverse forms from multiple sources. The lack of ability to deal with different types of constraints limits the generalization ability of these methods.

Concerning the integration of different types of constraints into the semi-supervised clustering methods, earlier methods [7, 26] discuss them case by case with different algorithms. However, these methods lack a unified view of constraints and are unable to deal with mixed types of constraints. Bai *et al.* resolved this issue via a unified formulation of pairwise constraints and label constraints [1] and proposed the SC-MPI algorithm to optimize them simultaneously. However, SC-MPI, which is designed for partitioning clustering, cannot perform hierarchical clustering and thus has limited generalization ability. Hierarchical clustering does not require specifying the number of clusters in advance, and it produces a dendrogram that shows the nested structure of the data. This is useful for many applications, such as finding cell subtypes in biological data analysis [5].

To address aforementioned issues, we propose a more general Semi-supervised clustering method via Structural Entropy with different constraints, namely SSE, for both partitioning clustering and hierarchical clustering. First, we construct a data graph G and a relation graph G' sharing the same set of vertices to represent the information of input data and prior infor-

*SKLSDE, Beihang University. {zengguangjie, penghao, angsheng, yangrunze}@buaa.edu.cn
†Salesforce AI Research. zhiweiliu@salesforce.com
‡Didi Chuxing. liuchunyang@didiglobal.com
§Lehigh University. lih319@lehigh.edu

mation in constraints, respectively. Vertices and edge weights of G are data points and similarities between them, respectively. Different types of constraints are formulated as a uniform view and stored in G' with positive edge weights representing must-link relationships and negative weights representing cannot-link relationships between data points. Second, we devise the objective of two-dimensional (2-d) SSE for semi-supervised partitioning clustering by adding a penalty term to the objective of 2-d structural entropy and then optimize it through two modified operators *merging* and *moving*. Third, we devise the objective of high-dimensional (high-d) SSE for semi-supervised hierarchical clustering by extending the objective of 2-d SSE, and then optimize it through two modified operators *stretching* and *compressing*. A binary encoding tree is obtained by *stretching* and an encoding tree with a certain height is obtained by *compressing*. The source code is available on GitHub[1].

We comprehensively evaluate SSE regarding semi-supervised clustering methods with respect to two types of constraints. The results justify the better performance of SSE under both types of constraints. We also conduct experiments on four single-cell RNA-seq datasets to perform cell clustering, demonstrating the functionality of SSE for biological data analysis. The main contributions of this paper are summarized as follows: (1) We devise a uniform formulation for pairwise constraints and label constraints and use them in a penalty term to form the objective of SSE. (2) We design efficient algorithms to optimize the objective of SSE to enable semi-supervised partitioning clustering and hierarchical clustering. (3) The extensive experiments on nine clustering datasets and four single-cell RNA-seq datasets indicate that SSE achieves the best performance among semi-supervised clustering methods and is effective for biological data analysis.

2 Structural Entropy

We provide a brief introduction to structural entropy [15] before presenting our model. Intuitively, structural entropy methods encode tree structures via characterizing the uncertainty of the hierarchical topology. The structural entropy of a graph G is defined as the minimum total number of bits required to determine the codewords of nodes in G. Structural entropy has achieved success in the field of traffic forecast [32], social event detection [3], and reinforcement learning [29, 30]. Through minimizing the structural entropy of a given graph G, the hierarchical clustering result of vertices in G is retained by the associated encoding tree.

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

Encoding tree. 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 as a hierarchical rooted tree is defined as follows: (1) For each tree node $\alpha \in \mathcal{T}$, a vertex subset $T_\alpha \in V$ is associated with it. (2) The root node λ of $\mathcal{T}$ is associated with the vertex set V, i.e., $T_\lambda = V$. (3) For each $\alpha \in \mathcal{T}$, the immediate successors of it are labeled by $\alpha^\wedge\langle i\rangle$ ordered from left to right as i increases, and the immediate predecessor of it is written as α^-. (4) For each $\alpha \in \mathcal{T}$ with L immediate successors, vertex subsets $T_{\alpha^\wedge\langle i\rangle}$ are disjoint and $T_\alpha = \cup_{i=1}^{L} T_{\alpha^\wedge\langle i\rangle}$. (5) For each leaf node $\nu \in \mathcal{T}$, T_ν contains only one vertex in V.

K-D Structural Entropy. Given an arbitrary rooted encoding tree $\mathcal{T}$ of a graph G, the structural entropy of G on $\mathcal{T}$ measures the amount of remaining complexity in G after reduced by $\mathcal{T}$. For each non-root node $\alpha \in \mathcal{T}$, the assigned structural entropy of it is defined as:

$$\mathcal{H}^{\mathcal{T}}(G;\alpha) = -\frac{g_\alpha}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\alpha}{\mathcal{V}_{\alpha^-}}, \tag{2.1}$$

where g_α is the cut, i.e., the weight sum of edges between nodes in and not in T_α, $\mathcal{V}_\alpha$ and $\mathcal{V}_G$ are the volumes, i.e., the sum of node degrees in T_α and G, respectively. The structural entropy of G given by $\mathcal{T}$ is defined as:

$$\mathcal{H}^{\mathcal{T}}(G) = \sum_{\alpha\in\mathcal{T},\alpha\neq\lambda}\mathcal{H}^{\mathcal{T}}(G;\alpha). \tag{2.2}$$

To meet the requirements of downstream applications, the K-dimensional structural entropy of G is defined as:

$$\mathcal{H}^{K}(G) = \min_{\mathcal{T}}\{\mathcal{H}^{\mathcal{T}}(G)\}, \tag{2.3}$$

where $\mathcal{T}$ ranges over all encoding trees whose heights are at most K.

2-D Structural Entropy. One special case of K-d structural entropy is 2-d structural entropy, where the encoding tree represents a graph partitioning. 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:

$$\begin{aligned}\mathcal{H}^{\mathcal{P}}(G) = &-\sum_{X\in\mathcal{P}}\sum_{v_i\in X}\frac{g_i}{\mathcal{V}_G}\log_2\frac{d_i}{\mathcal{V}_X}\\ &-\sum_{X\in\mathcal{P}}\frac{g_X}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_X}{\mathcal{V}_G},\end{aligned} \tag{2.4}$$

where d_i is the degree of vertex v_i, g_i is the cut, i.e., the weight sum of edges connecting v_i and other vertices, $\mathcal{V}_X$

and $\mathcal{V}_G$ are the volumes, i.e., the sum of node degrees in module X and graph G, respectively, and g_X is the cut, i.e., the weight sum of edges between vertices in and not in module X.

3 Methodology

In this section, we present the proposed SSE algorithm for semi-supervised clustering. The framework of SSE is depicted in Figure 1. SSE has three components: graph construction, semi-supervised partitioning clustering, and semi-supervised hierarchical clustering. Input data and constraints are transformed into two different graphs sharing the same vertex set and then used to perform semi-supervised partitioning clustering and semi-supervised hierarchical clustering through 2-d SSE and high-d SSE minimization, respectively.

3.1 Uniform Formulation of Constraints. Considering a graph $G = (V, E, \mathbf{W})$ associated to a given dataset $\mathcal{X} = \{x_1, x_2, ..., x_n\}$, where x_i is a data point, $V = \{v_1, v_2, ..., v_n\}$ correspond to data points in $\mathcal{X}$, the edges in E connect similar data points, and edge weights in $\mathbf{W}$ represent similarity of data points. We aim to partition graph vertices in G with certain given prior information in the form of constraints to achieve semi-supervised data clustering. The pairwise constraints and label constraints are formulated as follows.

Pairwise constraints reveal the relationship between a pair of vertices in G. They consist of a set of must-link constraints $M = \{(v_i, v_j) \colon l_i = l_j\}$, indicating that vertex pair (v_i, v_j) must belong to the same cluster, and a set of cannot-link constraints $C = \{(x_i, x_j) \colon l_i \neq l_j\}$, indicating that vertex pair (v_i, v_j) must belong to different clusters, where l_i is the cluster indicator of v_i. Pairwise constraints can be stored in a relation graph $G' = (V, E', \mathbf{W}')$, which shares the same vertex set with G. If there exists a vertex pair $(v_i, v_j) \in M$, an edge exists in E' with a positive value γ_M added to the edge weight $\mathbf{W}'_{ij}$. If there exists a vertex pair $(v_i, v_j) \in C$, an edge exists in E' with a negative value γ_C added to the edge weight $\mathbf{W}'_{ij}$. The values of $\mathbf{W}$ and $\mathbf{W}'$ are set in **Implementation Details** in Section 4.

Label constraints reveal the relationship between vertices in G and ground truth class labels. They include a set of positive-label constraints $P = \{(v_i, y_m) \colon v_i \in y_m\}$, indicating that the true class label of v_i is y_m, and a set of negative-label constraints $N = \{(v_i, y_m) \colon v_i \notin y_m\}$, indicating that the true class label of v_i is not y_m. To form a uniform representation of constraints, we convert label constraints into pairwise constraints which are more compatible for structural entropy. For two vertices v_i and v_j, the conversion rules are set as follows: (1) If they both 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 both 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 constraints respectively with the same label, an edge exists in E' with a negative value γ_C added to the edge weight $\mathbf{W}'_{ij}$.

The constraints are stored in the relation graph G' after construction, where a positive value indicates a must-link relationship and a negative value indicates a cannot-link relationship. However, this relation graph can be further improved by exploiting *constraint transitivity* and *entailment* [23]. We apply them sequentially on G' after constructing it.

3.2 2-D SSE. In this subsection, we present 2-d SSE modified from 2-d structural entropy to perform semi-supervised partitioning clustering. For a graph G associated with a data set $\mathcal{X}$ with different types of constraints, we transform all types of constraints into a uniform formulation and store them in a relation graph G'. We aim to find a graph partitioning $\mathcal{P}$ of G that minimizes the structural entropy of G while also minimizing the number of violated constraints in the meantime. The optimization objective of two-dimensional structural entropy is defined as follows:

$$\mathcal{L}^{\mathcal{P}}(G, G') = \mathcal{H}^{\mathcal{P}}(G) + \phi \mathcal{E}^{\mathcal{P}}(G, G'), \tag{3.5}$$

where $\mathcal{E}^{\mathcal{P}}(G, G')$ is a penalty term for constraints violation, and it is defined as:

$$\mathcal{E}^{\mathcal{P}}(G, G') = -\sum_{X \in \mathcal{P}} \frac{g'_X}{\mathcal{V}_G} \log_2 \frac{\mathcal{V}_X}{\mathcal{V}_G}, \tag{3.6}$$

where g'_X is the weight sum of edges in G' between vertices in and not in module X, and other notations share the same meaning with notations in Eq. (2.4).

The intuition of the penalty term is that we modify g_X, i.e., the cut of module X in Eq. (2.4) 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 module, 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 modules, 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 SSE. We minimize 2-d SSE via two operators *merging* [15] and *moving* on the encoding tree

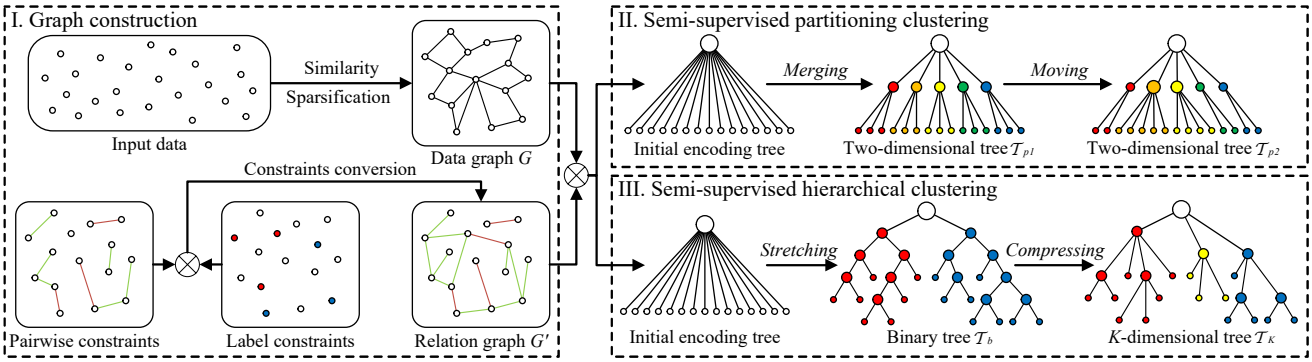

Figure 1: Overview of SSE. (I) Two graphs G and G' are constructed from input data and constraints, respectively. (II) Semi-supervised partitioning clustering is performed through two operators *merging* and *moving*. (III) Semi-supervised hierarchical clustering is performed through two operators *stretching* and *compressing*.

$\mathcal{T}$. For two sister nodes $\alpha, \beta \in \mathcal{T}$ with associated vertex subsets X and Y, node *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:

$$\begin{aligned}\Delta\mathcal{L}^{\mathcal{M}}_{X,Y} = \frac{1}{\mathcal{V}_G}[&(\mathcal{V}_X - g_X - g'_X)\log_2 \mathcal{V}_X \\ &+ (\mathcal{V}_Y - g_Y - g'_Y)\log_2 \mathcal{V}_Y \\ &- (\mathcal{V}_{X\cup Y} - g_{X\cup Y} - g'_{X\cup Y})\log_2 \mathcal{V}_{X\cup Y} \\ &+ (g_X + g_Y - g_{X\cup Y} + g'_X + g'_Y - g'_{X\cup Y})\log_2 \mathcal{V}_G],\end{aligned} \tag{3.7}$$

where $\mathcal{M}$ denotes $\mathcal{M}$erging operator, $\mathcal{V}_X$ 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. For a node $\alpha \in \mathcal{T}$ with associated module X and a vertex $v_i \in X$, the *moving* operator seeks to find a new node $\beta \in \mathcal{T}$ with associated module Y and move v_i from X to Y. The decrease amount of $\mathcal{L}^{\mathcal{P}}(G, G')$ by removing v_i from X is given by:

$$\begin{aligned}\Delta\mathcal{L}^{\mathcal{R}}_{X,v_i} = &\frac{\mathcal{V}_X - g_X - g'_X}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_X}{\mathcal{V}_G} \\ -&\frac{\mathcal{V}_{X\setminus\{v_i\}} - g_{X\setminus\{v_i\}} - g'_{X\setminus\{v_i\}}}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_{X\setminus\{v_i\}}}{\mathcal{V}_G},\end{aligned} \tag{3.8}$$

where $\mathcal{R}$ denotes vertex $\mathcal{R}$emoving 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:

$$\begin{aligned}\Delta\mathcal{L}^{\mathcal{I}}_{Y,v_i} = &-\frac{\mathcal{V}_Y - g_Y - g'_Y}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_Y}{\mathcal{V}_G} \\ +&\frac{\mathcal{V}_{Y\cup\{v_i\}} - g_{Y\cup\{v_i\}} - g'_{Y\cup\{v_i\}}}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_{Y\cup\{v_i\}}}{\mathcal{V}_G},\end{aligned} \tag{3.9}$$

where $\mathcal{I}$ denotes vertex $\mathcal{I}$nserting 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.

Algorithm 1 2-d SSE minimization

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

```
Initialize T containing all vertices as tree leaves
// Merging stage
repeat
    Merge a chosen module pair (X, Y) into X ∪ Y condition on arg max_{X,Y}{ΔL^M_{X,Y}} via Eq. (3.7)
    Update ΔL^M for module pairs connected to X or Y
until ΔL^M < 0 for all module pairs
// Moving stage
repeat
    for each vertex v_i ∈ V do
        Remove vertex v_i from the original module X
        Insert node v_i into a chosen module Y condition on arg max_Y{ΔL^R_{X,v_i} − ΔL^I_{Y,v_i}} via Eqs. (3.8) and (3.9)
    end for
until L^P(G, G') converges
```

In both *merging* stage and *moving* stage, $\mathcal{L}^{\mathcal{P}}$ decreases after every iteration, and it converges when no improvement can be made. The time complexity of *merging* stage is $O(n\log^2 n)$ [15]. 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 $O(nl)$. Taken together, the time complexity of Algorithm 1 is $O(n\log^2 n + nlt)$, where n, l and t denote the number of vertices, modules, and iterations respectively.

3.3 High-D SSE. Hereafter, we generalize 2-d SSE into high-d SSE to perform semi-supervised hierarchical clustering. For a graph G associated with data set $\mathcal{X}$ and constraints stored in a relation graph G', we aim to find an encoding tree with the height of $K > 2$ to form a semi-supervised hierarchical clustering of vertices in G. Following the definition of 2-d SSE in Section 3.2, we define the optimization objective of high-d SSE as follows:

$$\mathcal{L}^{\mathcal{T}}(G, G') = \mathcal{H}^{\mathcal{T}}(G) + \phi\mathcal{E}^{\mathcal{T}}(G, G'), \tag{3.10}$$

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

$$\mathcal{E}^{\mathcal{T}}(G, G') = \sum_{\alpha\in\mathcal{T}, 1<|T(\alpha)|<|V|} -\frac{g'_\alpha}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\alpha}{\mathcal{V}_{\alpha^-}}, \tag{3.11}$$

where g'_α is the cut of α in G', $|T(\alpha)|$ is the number of vertices in subset $T(\alpha)$ associated to α, and other notations share the same meaning with notations in Eq. (2.1). For each node except for leaves in $\mathcal{T}$, the penalty term penalizes the violation of must-link constraints and rewards the satisfaction of cannot-link constraints.

Minimizing High-D SSE. We minimize high-d SSE via two operators *stretching* and *compressing* on the encoding tree $\mathcal{T}$ [19]. 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:

$$\Delta\mathcal{L}^{\mathcal{S}}_{\alpha,\beta} = \frac{g_\alpha + g_\beta - g_\delta + g'_\alpha + g'_\beta - g'_\delta}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\gamma}{\mathcal{V}_\delta}, \tag{3.12}$$

where $\mathcal{S}$ denotes node $\mathcal{S}$tretching. Applying *stretching* on the initial encoding tree iteratively results in a binary encoding tree $\mathcal{T}_b$. 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:

$$\Delta\mathcal{L}^{\mathcal{C}}_{\alpha} = \frac{\sum_i g_{\beta_i} + \sum_{|T(\beta_i)|>1} g'_{\beta_i} - g_\alpha - g'_\alpha}{\mathcal{V}_G}\log_2\frac{\mathcal{V}_\alpha}{\mathcal{V}_\gamma}, \tag{3.13}$$

where $\mathcal{C}$ denotes node $\mathcal{C}$ompressing. 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. For a graph G with n vertices and m edges, the time complexity of Algorithm 2 is $O(h_{max}(mlogn + n))$, where h_{max} is the height of $\mathcal{T}_b$.

Algorithm 2 High-d SSE minimization

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. (3.12)
  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. (3.13)
until Height of encoding tree $\mathcal{T}$ is not larger than K, resulting in $\mathcal{T}_K$

4 Experiments

Our proposed SSE method is capable of tackling both semi-supervised partitioning clustering and hierarchical clustering. Regarding the evaluation for both tasks, we design two groups of experiments, in which we compare SSE against established baselines for semi-supervised partitioning clustering (Section 4.1) and semi-supervised hierarchical clustering (Section 4.2).

4.1 Semi-Supervised Partitioning Clustering. In this part, we aim to evaluate the performance of SSE on semi-supervised partitioning clustering. We conduct experiments on five clustering datasets including face image data (Yale and ORL), object image data (COIL20), spoken letter recognition data (Isolet), and handwritten digit data (OpticalDigits) following Bai *et al.* [1], whose size ranges from 165 to 5620. We also conduct experiments on four single-cell RAN-seq datasets including 10X PBMC, Mouse bladder, Worm neuron, and Human kidney taken from Tian *et al.* [22]. We choose the data preprocessed by the original authors to contain 2100 randomly sampled cells on the top 2000 highly dispersed genes in each dataset. We adopt two metrics including Adjusted Rand Index (ARI) [10] and Normalized Mutual Information (NMI) [21] for partitioning clustering performance evaluation. All experiments are repeated 10 times.

Baselines. We compare SSE with a variety of baseline methods, including an unsupervised clustering method based on structural entropy minimization, three semi-supervised clustering methods with pairwise constraints, three semi-supervised clustering methods with

Table 1: Performance of comparison methods for partitioning clustering on five clustering datasets. **Bold**: the best performance on each group of methods.

	Method%	Yale		ORL		COIL20		Isolet		OpticalDigits	
		ARI↑	NMI↑	ARI↑	NMI↑	ARI↑	NMI↑	ARI↑	NMI↑	ARI↑	NMI↑
	SE	28.12	54.78	59.15	85.31	68.22	86.34	56.01	83.14	69.89	79.69
Pairwise	PCPSNMF	24.80	54.11	52.02	81.86	51.49	80.75	38.39	69.15	48.82	68.71
	OneStepPCP	25.50	52.22	40.58	78.14	52.81	79.70	49.93	76.01	77.90	86.02
	CMS	07.06	35.54	29.37	73.18	59.81	78.32	48.77	77.38	**88.75**	**91.17**
	SC-MPI	32.76	59.82	49.28	82.29	59.89	82.83	47.16	72.75	52.39	71.35
	SSE (Ours)	**37.12**	**61.37**	**65.42**	**87.51**	**75.36**	**87.50**	**61.37**	**82.77**	77.57	84.34
Label	Seeded-KMeans	25.21	52.06	46.35	78.56	67.59	81.40	**66.48**	81.13	73.52	77.89
	S4NMF	23.85	49.60	47.23	77.08	62.48	79.34	57.05	77.32	84.72	88.32
	LpCNMF	13.35	39.67	32.55	70.01	74.72	88.78	59.24	81.89	90.77	**93.80**
	SC-MPI	20.91	50.82	26.28	70.73	**89.18**	**94.21**	64.43	80.38	**93.04**	93.41
	SSE (Ours)	**33.48**	**58.62**	**61.26**	**86.01**	75.10	87.63	58.62	**83.13**	76.60	84.12

Table 2: Performance of comparison methods for partitioning clustering on four RNA-seq datasets. **Bold**: the best performance on each group of methods.

	Method%	10X PBMC		Mouse bladder		Worm neuron		Human kidney	
		ARI↑	NMI↑	ARI↑	NMI↑	ARI↑	NMI↑	ARI↑	NMI↑
	SE	63.89	74.80	67.41	77.13	20.90	41.70	54.20	72.86
Pairwise	PCPSNMF	16.41	29.68	13.55	40.02	09.45	21.48	13.46	28.66
	OneStepPCP	43.08	58.29	44.51	64.86	19.39	44.42	40.21	55.26
	CMS	08.58	10.40	08.28	10.56	00.27	01.40	07.18	12.26
	SC-MPI	20.24	30.57	18.70	40.88	08.98	17.03	18.12	31.78
	SSE (Ours)	**74.87**	**76.84**	**62.33**	**74.70**	**22.17**	**45.05**	**62.59**	**75.81**
Label	Seeded-KMeans	67.56	71.07	38.40	62.63	07.07	34.24	17.19	39.74
	S4NMF	18.28	28.20	26.27	43.79	08.90	15.54	24.33	39.07
	LpCNMF	44.40	62.94	44.64	73.02	34.67	**56.37**	45.90	64.88
	SC-MPI	48.28	63.34	38.65	55.23	**37.82**	46.58	56.93	60.36
	SSE (Ours)	**74.15**	**77.05**	**63.11**	**75.38**	28.43	46.94	**65.45**	**78.23**

label constraints, and one semi-supervised clustering method with both pairwise constraints and label constraints. The unsupervised clustering based on structural entropy minimization is optimized by the *merging* operator (SE [15]). For semi-supervised clustering methods with pairwise constraints, we consider pairwise constraint propagation-induced symmetric NMF (PCPSNMF [25]), jointly optimized pairwise constraint propagation and spectral clustering (OneStepPCP [11]), and constrained mean shift clustering (CMS [20]). For semi-supervised clustering methods with label constraints, we consider seeded semi-supervised KMeans (Seeded-KMeans [2]), self-supervised semi-supervised NMF (S4NMF [4]), and label propagation based constrained NMF (LpCNMF [17]). SC-MPI [1] is a semi-supervised spectral clustering method capable of dealing with different types of constraints. Since SSE and SC-MPI are capable of dealing with both pairwise constraints and label constraints, they are compared in both groups.

Implementation Details. We construct graph G from the given dataset $\mathcal{X}$ by calculating the similarity between data points and then sparsify it into a p-nearest-neighbor graph by retaining p most significant edges from each node. For a given $\mathcal{X}$ with n data points divided into k clusters according to the ground truth, we empirically set p to be $\lfloor 20k/\log_2^2 n \rfloor + 1$, since the number of clusters by minimizing $\mathcal{H}^{\mathcal{P}}$ is approximately $\Theta(p \log_2^2 n)$ [15]. For five clustering datasets, the similarity is defined by a Gaussian kernel with kernel width $\sigma = 10$. For four single-cell RNA-seq datasets, the similarity is defined as cosine similarity since the features of these datasets are sparse.

We generate constraints using the ground truth

Table 3: Performance of comparison methods for semi-supervised hierarchical clustering. **Bold**: the best performance on each group of methods.

Method%	Wine			Heart			Br. Cancer			Australian		
	DP↑	ARI↑	NMI↑	DP↑	ARI↑	NMI↑	DP↑	ARI↑	NMI↑	DP↑	ARI↑	NMI↑
SE	84.87	73.85	74.19	61.97	07.70	21.30	95.75	88.00	80.93	57.02	01.95	08.03
SpecWRSC	84.87	76.94	77.10	70.13	**33.13**	**29.23**	95.68	**88.55**	**81.57**	54.92	-00.78	01.21
COBRA	86.50	81.26	80.07	64.12	26.07	20.92	92.23	82.38	72.41	66.55	32.03	24.61
SemiMulticons	90.52	82.99	82.69	69.29	28.44	26.91	92.68	82.77	73.79	72.13	**39.47**	**33.83**
SSE (Ours)	**92.88**	**85.27**	**83.61**	**71.90**	28.08	26.36	**96.53**	82.88	76.08	**74.52**	34.19	28.17

class labels from the datasets. For experiments with pairwise constraints, we set the number of must-link constraints the same as cannot-link constraints to be $0.2n$. For experiments with label constraints, we set the number of positive constraints the same as negative constraints to be $0.1n$. The parameters γ_M and γ_C control the role of constraints, we define them following Bai *et al.* [1]. For a pair of data points (x_i, x_j) with similarity $\mathbf{W}_{ij}$, we define $\gamma_M = max(\mathbf{W}) - \mathbf{W}_{ij}$, where $max(\mathbf{W})$ is the maximum similarity between all data points. The process of constraints conversion, *constraints transitivity* and *entailment* usually lead to more negative values than positive values in G'. In order to balance them, we define $\gamma_C = \rho(min(\mathbf{W}) - \mathbf{W}_{ij})$, where ρ is the ratio between the number of positive values and negative values in G', $min(\mathbf{W})$ is the minimum similarity between all data points. The parameter ϕ balances the importance between input data and constraints, it is empirically set as $\phi = 2$.

Experimental Results. The experimental results on five clustering datasets are presented in Table 1. Three groups of methods, i.e., unsupervised clustering, semi-supervised clustering with pairwise constraints, and semi-supervised clustering with label constraints, are compared separately. SSE with pairwise constraints outperforms its unsupervised baseline SE on all datasets and outperforms baseline methods in the pairwise constraint group on four out of five datasets. SSE with label constraints outperforms SE on all datasets and outperforms baseline methods in the label constraint group on three out of five datasets.

The experimental results on four single-cell RNA-seq datasets are presented in Table 2. SSE with pairwise constraints outperforms SE on three out of four datasets and outperforms baseline methods in the pairwise constraint group on all datasets. SSE with label constraints outperforms SE on three out of four datasets and outperforms baseline methods in the label constraint group on three out of four datasets. In all, SSE effectively incorporates prior information in the forms of pairwise constraints and label constraints, and achieves high clustering accuracy on both clustering datasets and single-cell RNA-seq datasets.

4.2 Semi-Supervised Hierarchical Clustering.

In this part, we aim to evaluate the performance of SSE on semi-supervised hierarchical clustering. We conduct experiments on four datasets downloaded from the LIBSVM webpage [2] following Chierchia and Perret [6], whose size ranges from 175 to 690. We adopt three metrics including Dendrogram Purity (DP) [9,27], ARI [10], and NMI [21] for hierarchical clustering performance evaluation. DP is a holistic measure of a cluster tree, which is defined as the weighted average purity of each node of the tree with respect to a ground truth labelization of the tree leaves. We take the cluster tree of SSE from the binary encoding tree $\mathcal{T}_b$. ARI and NMI require partitioning clustering results from the cluster trees. We perform the *compressing* operator until the height of the encoding tree is two to obtain the partitioning clustering results. For other methods, we choose the largest tree nodes from the cluster tree as the partitioning clustering results. All experiments are repeated 10 times.

Baselines. We compare SSE with two unsupervised hierarchical clustering methods and two semi-supervised hierarchical clustering methods. For unsupervised hierarchical clustering methods, we consider structural entropy minimized by *stretching* operator and *compressing* operator (SE [19]) and sublinear time graph-based hierarchical clustering (SpecWRSC [13]). For semi-supervised hierarchical clustering methods, we consider merging-based active clustering (COBRA [23]) and closed pattern mining based semi-supervised consensus clustering (SemiMulticons [28]).

Implementation Details. We construct graph G from the given dataset $\mathcal{X}$ by calculating cosine similarity between data points and then sparsify it into a 5-nearest-neighbor graph by retaining 5 significant edges from each node. We generate $0.2n$ must-link constraints and $0.2n$ cannot-link constraints randomly for all meth-

[2] https://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/

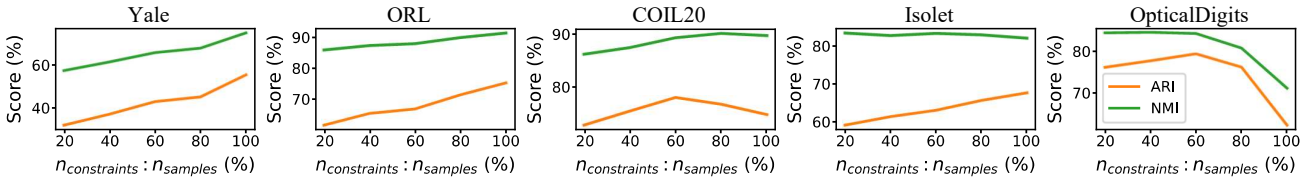

Figure 2: Performance of SSE for semi-supervised partitioning clustering with different constraint amounts.

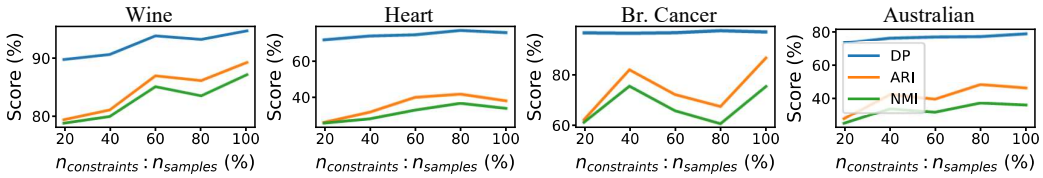

Figure 3: Performance of SSE for semi-supervised hierarchical clustering with different constraint amounts.

ods except for COBRA, for which we generate $0.2n$ positive constraints due to its requirements. We set $\gamma_M = max(\mathbf{W})\mathbf{W}_{ij}$ and $\gamma_C = \rho(min(\mathbf{W})\mathbf{W}_{ij})$. The parameter ϕ is set to be 2.

Experimental Results. The experimental results of semi-supervised hierarchical clustering are presented in Table 3. SSE achieves the highest DP values and outperforms SE in terms of DP on all datasets, indicating that the cluster trees of SSE have the highest holistic quality. The ARI and NMI values of SSE are comparable with baselines and higher than SE on three out of four datasets. In all, SSE achieves high clustering accuracy on semi-supervised hierarchical clustering.

4.3 Sensitivity Analysis. The number of constraints has a great impact on the performance of semi-supervised clustering and a larger number of constraints is usually thought to lead to better performance. We evaluate the performance of SSE for partitioning clustering with different amounts of pairwise constraint, as shown in Figure 2. The ARI and NMI values are generally larger with more constraints except for OpticalDigits. For this dataset, SSE makes too many clusters when the constraints are more than $0.6n$, leading to poor performance. The cause of this problem is that the *merging* stage in Algorithm 1 stops earlier than expected, which calls for a better optimization algorithm. We also evaluate the performance of SSE for hierarchical clustering with different amounts of pairwise constraint, as shown in Figure 3. The growth of DP values can be barely seen, since the DP values of all amounts of constraint are very high. The ARI and NMI values grow a lot with more constraints provided. In all, SSE performs better with more constraints under most circumstances.

5 Related Work

Semi-supervised clustering methods incorporate prior information into the process of clustering to enhance clustering quality and better align user preferences, and have attracted great interest in recent years. Prior information can take different forms of constraints, among them pairwise constraints and label constraints are mostly used. Pairwise constraints indicate whether a pair of data points should be in the same cluster or not [18]. Many methods that incorporate pairwise constraints have been proposed, such as semi-supervised spectral clustering [1], semi-supervised NMF clustering [25], and semi-supervised density peak clustering [20]. Label constraints reveal class labels of some data points, specifying whether they belong to certain classes or not. These constraints can be used through label propagation [17] or penalizing violated data points [4].

6 Conclusion

In this paper, we propose SSE, a novel and more general semi-supervised clustering method that can integrate different types of constraints. We give a uniform formulation of pairwise constraints and label constraints and make them both compatible with SSE. Moreover, SSE can perform both semi-supervised partitioning clustering and hierarchical clustering, thanks to the structural entropy measure that it is based on. We conduct extensive experiments on nine clustering datasets and compare SSE with eleven baselines, justifying the superiority of SSE on high clustering accuracy. We also apply SSE to four single-cell RNA-seq datasets for cell clustering, demonstrating its functionality in biological data analysis. Future work on SSE may focus on better optimization algorithms.

Acknowledgments

The corresponding author is Hao Peng. This work is supported by National Key R&D Program of China through grant 2021YFB1714800, NSFC through grants 61932002 and 62322202, Beijing Natural Science Foundation through grant 4222030, CCF-DiDi GAIA Collaborative Research Funds for Young Scholars, and the Fundamental Research Funds for the Central Universities.

References

[1] L. Bai, J. Liang, and F. Cao, *Semi-supervised clustering with constraints of different types from multiple information sources*, IEEE TPAMI, 43 (2020), pp. 3247–3258.

[2] S. Basu, A. Banerjee, and R. J. Mooney, *Semi-supervised clustering by seeding*, in ICML, 2002, pp. 19–26.

[3] Y. Cao, H. Peng, Z. Yu, and Y. Philip S., *Hierarchical and incremental structural entropy minimization for unsupervised social event detection*, in AAAI, 2024, pp. 1–10.

[4] J. Chavoshinejad, S. A. Seyedi, F. A. Tab, and N. Salahian, *Self-supervised semi-supervised nonnegative matrix factorization for data clustering*, Pattern Recognition, 137 (2023), p. 109282.

[5] L. Chen and S. Li, *Incorporating cell hierarchy to decipher the functional diversity of single cells*, Nucleic Acids Research, 51 (2022), pp. e9–e9.

[6] G. Chierchia and B. Perret, *Ultrametric fitting by gradient descent*, in NeurIPS, vol. 32, 2019.

[7] I. Davidson and S. Ravi, *Agglomerative hierarchical clustering with constraints: Theoretical and empirical results*, in ECML PKDD, Springer, 2005, pp. 59–70.

[8] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*, SIAM, 2020.

[9] K. A. Heller and Z. Ghahramani, *Bayesian hierarchical clustering*, in ICML, 2005, pp. 297–304.

[10] L. Hubert and P. Arabie, *Comparing partitions*, Journal of classification, 2 (1985), pp. 193–218.

[11] Y. Jia, W. Wu, R. Wang, J. Hou, and S. Kwong, *Joint optimization for pairwise constraint propagation*, IEEE TNNLS, 32 (2020), pp. 3168–3180.

[12] Z. Jiang, Y. Zhan, Q. Mao, and Y. Du, *Semi-supervised clustering under a "compact-cluster" assumption*, IEEE TKDE, 35 (2022), pp. 5244–5256.

[13] M. Kapralov, A. Kumar, S. Lattanzi, and A. Mousavifar, *Learning hierarchical cluster structure of graphs in sublinear time*, in SODA, SIAM, 2023, pp. 925–939.

[14] L. Lan, T. Liu, X. Zhang, C. Xu, and Z. Luo, *Label propagated nonnegative matrix factorization for clustering*, IEEE TKDE, 34 (2020), pp. 340–351.

[15] A. Li and Y. Pan, *Structural information and dynamical complexity of networks*, IEEE Transactions on Information Theory, 62 (2016), pp. 3290–3339.

[16] J. Lipor and L. Balzano, *Leveraging union of subspace structure to improve constrained clustering*, in ICML, PMLR, 2017, pp. 2130–2139.

[17] J. Liu, Y. Wang, J. Ma, D. Han, and Y. Huang, *Constrained nonnegative matrix factorization based on label propagation for data representation*, IEEE TAI, (2023).

[18] F. Nie, H. Zhang, R. Wang, and X. Li, *Semi-supervised clustering via pairwise constrained optimal graph*, in IJCAI, 2021, pp. 3160–3166.

[19] Y. Pan, F. Zheng, and B. Fan, *An information-theoretic perspective of hierarchical clustering*, arXiv preprint arXiv:2108.06036, (2021).

[20] M. Schier, C. Reinders, and B. Rosenhahn, *Constrained mean shift clustering*, in SDM, SIAM, 2022, pp. 235–243.

[21] A. Strehl and J. Ghosh, *Cluster ensembles—a knowledge reuse framework for combining multiple partitions*, JMLR, 3 (2002), pp. 583–617.

[22] 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*, Nature communications, 12 (2021), p. 1873.

[23] T. Van Craenendonck, S. Dumančic, and H. Blockeel, *Cobra: a fast and simple method for active clustering with pairwise constraints*, in IJCAI, 2017, pp. 2871–2877.

[24] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, et al., *Constrained k-means clustering with background knowledge*, in ICML, vol. 1, 2001, pp. 577–584.

[25] W. Wu, Y. Jia, S. Kwong, and J. Hou, *Pairwise constraint propagation-induced symmetric nonnegative matrix factorization*, IEEE TNNLS, 29 (2018), pp. 6348–6361.

[26] W. Xiao, Y. Yang, H. Wang, T. Li, and H. Xing, *Semi-supervised hierarchical clustering ensemble and its application*, Neurocomputing, 173 (2016), pp. 1362–1376.

[27] N. Yadav, A. Kobren, N. Monath, and A. Mccallum, *Supervised hierarchical clustering with exponential linkage*, in ICML, PMLR, 2019, pp. 6973–6983.

[28] T. Yang, N. Pasquier, and F. Precioso, *Semi-supervised consensus clustering based on closed patterns*, Knowledge-Based Systems, 235 (2022), p. 107599.

[29] X. Zeng, H. Peng, and A. Li, *Effective and stable role-based multi-agent collaboration by structural information principles*, in AAAI, 2023.

[30] X. Zeng, H. Peng, and A. Li, *Adversarial socialbots modeling based on structural information principles*, in AAAI, 2024, pp. 1–10.

[31] L. Zheng and T. Li, *Semi-supervised hierarchical clustering*, in IEEE ICDM, 2011, pp. 982–991.

[32] D. Zou, S. Wang, X. Li, H. Peng, Y. Wang, C. Liu, K. Sheng, and B. Zhang, *Multispans: A multi-range spatial-temporal transformer network for traffic forecast via structural entropy optimization*, in ACM WSDM, 2024, pp. 1–10.