Optimized Algorithms for Text Clustering with LLM-Generated Constraints

Chaoqi Jia[1], Weihong Wu[2], Longkun Guo[2*], Zhigang Lu[3], Chao Chen[1], Kok-Leong Ong[1]

[1] School of Accounting, Information Systems and Supply Chain, RMIT University, Melbourne, VIC 3000, Australia
[2] School of Mathematics and Statistics, Fuzhou University, Fuzhou 350116, China
[3] Western Sydney University, NSW 2751, Australia
chaoqi.jia@student.rmit.edu.au, weihong.research@gmail.com, longkun.guo@gmail.com, z.lu@westernsydney.edu.au, {chao.chen, kok-leong.ong2}@rmit.edu.au

Abstract

Clustering is a fundamental tool that has garnered significant interest across a wide range of applications including text analysis. To improve clustering accuracy, many researchers have incorporated background knowledge, typically in the form of must-link and cannot-link constraints, to guide the clustering process. With the recent advent of large language models (LLMs), there is growing interest in improving clustering quality through LLM-based automatic constraint generation. In this paper, we propose a novel constraint-generation approach that reduces resource consumption by generating constraint sets rather than using traditional pairwise constraints. This approach improves both query efficiency and constraint accuracy compared to state-of-the-art methods. We further introduce a constrained clustering algorithm tailored to the characteristics of LLM-generated constraints. Our method incorporates a confidence threshold and a penalty mechanism to address potentially inaccurate constraints. We evaluate our approach on five text datasets, considering both the cost of constraint generation and the overall clustering performance. The results show that our method achieves clustering accuracy comparable to the state-of-the-art algorithms while reducing the number of LLM queries by more than 20 times.

Code — https://github.com/weihong-wu/LSCK-HC

1 Introduction

Short text clustering (STC) is a well-known natural language processing task, widely used for analyzing brief user-generated content on platforms such as Twitter, Instagram, and Facebook (Ahmed et al. 2022). Thanks to its simplicity and efficiency, the k-means algorithm is widely applied when implementing the STC tasks. To further meet clients' expectations and improve the clustering quality, people have introduced semi-supervised approaches using additional background knowledge of the raw data to assist the k-means clustering (Cai et al. 2023).

For text clustering with pairwise constraints, many researchers (Basu, Banerjee, and Mooney 2002; Bae et al. 2020; Viswanathan et al. 2024) have incorporated background knowledge into k-means clustering. For instance, Wagstaff and Cardie (2000) introduced the concepts of cannot-link (CL) and must-link (ML) constraints and designed a clustering algorithm that enforces these constraints. Building upon this work, Basu, Banerjee, and Mooney (2002) developed algorithms that applied these constraints to textual data, such as news articles, utilizing spherical k-means for clustering. Following this, Basu, Banerjee, and Mooney (2004) proposed an active-learning approach to generate constraints automatically, thereby reducing reliance on manual input, and integrated these constraints directly into the clustering objective. However, these methods still depend on constraints that are either hand-picked by experts (Basu, Banerjee, and Mooney 2002) or derived from existing labels (Baumann and Hochbaum 2024). In textual datasets and unlabelled domains, such as social media text, manually generating a sufficient number of constraints is prohibitively time-consuming and costly.

Recently, LLMs have been considered to serve many different short text clustering tasks, for instance, they can be used to generate pairwise constraints for clustering by In-Context Learning (Zhang, Wang, and Shang 2023; Viswanathan et al. 2024) and text summarization (Feng et al. 2025; Zhang, Yu, and Zhang 2025). However, Viswanathan et al. (2024) incurred high costs while achieving limited improvements in constrained k-means clustering when applying their LLM-generated constraints. To fill the gap, we argue that LLMs can offer greater benefits if the pairwise constraint format is replaced with a set-based formulation and if the clustering algorithm is designed to tolerate false in LLM-generated constraints. Based on these challenges, we summarize our main contributions as follows.

- By leveraging the generation capabilities of LLMs, we provide a query selection method for decision-making to generate the CL and ML constraints. We then apply the LLMs to measure the thresholds to categorize the must-links as hard or soft constraints, which are used in the constrained clustering process.
- Based on the LLM-generated constraints, we propose a penalty-augmented local search clustering algorithm to handle the CL and ML constraints. Cannot-link constraints are handled via a local search with the penalty matching method, while must-link constraints are divided into two types: hard constraints guide the initial seeding represented by the mean of the set, and soft con-

*Corresponding Author

straints correct the clustering via penalties.

- By evaluating the LLM-generated clustering framework on 5 real-world text datasets, our results show that the clustering quality attains accuracy that is comparable to or exceeds that of the existing algorithms, while reducing LLM queries by more than 20 times.

2 Related Works

Clustering algorithms, such as k-means (Lloyd 1982) and k-means++ (Arthur and Vassilvitskii 2007), are widely used in short-text clustering. Extensive research has been conducted on interactive approaches that integrate partial supervision through must-link and cannot-link constraints (Bae et al. 2020). In this work, we concentrate on introducing two critical components of constrained clustering: constraint generation and clustering algorithms.

Generated Constraints. Basu, Banerjee, and Mooney (2004) first proposed an automated strategy for generating must-link and cannot-link constraints, known as farthest-first query selection, which reduces reliance on manual labeling. Then, other typical methods have been introduced, such as the min-max approach (Mallapragada, Jin, and Jain 2008) and normalized point-based uncertainty (Xiong, Azimi, and Fern 2013). The primary objective of these methods is to select the most informative and representative examples for supervision (Fu et al. 2024). In recent years, active clustering with pairwise constraints has been extensively adopted in semi-supervised clustering (Yu et al. 2018; Xiong, Johnson, and Corso 2016; Li et al. 2019; Fu et al. 2024). However, these approaches still rely on expert judgment or are derived from existing labels as prior knowledge. Recent works (Zhang, Wang, and Shang 2023; Viswanathan et al. 2024) have explored the use of In-Context Learning (ICL) with LLMs to generate pairwise constraints, while Huang and He (2024) employ LLMs to produce potential labels for classification tasks. In our work, we utilize the characteristics of LLMs and then design algorithms to enhance query efficiency by extending constraints from individual pairs to sets, thereby reducing the number of LLM queries required while improving the constraints' accuracy.

Constrained Clustering Algorithms. Wagstaff et al. (2001) incorporated must-link and cannot-link constraints into k-means clustering and proposed the COP-KMeans algorithm, a greedy approach designed for constraint satisfaction. Next, constrained clustering algorithms have typically categorized relationships between instances as either hard constraints which must always be satisfied (Basu, Banerjee, and Mooney 2002; Ganji, Bailey, and Stuckey 2016; Le et al. 2018; Baumann 2020; Jia et al. 2023), or soft constraints which are integrated into the objective function with penalties for violations (Basu, Banerjee, and Mooney 2004; Davidson and Ravi 2005b,a; Pelleg and Baras 2007; Baumann and Hochbaum 2022). More recently, Baumann and Hochbaum (2024) proposed the constraints clustering algorithm utilizing integer programming to cluster and effectively manage constraints allowing for the inclusion of both hard and soft types. In addition, Jia et al. (2023) and Guo et al. (2024) proposed approaches that handle constraints as disjoint sets to prove the approximation ratio of clustering. However, existing algorithms are not specifically designed to leverage the properties of LLM-generated constraints, even though soft-constraint clustering algorithms can mitigate some misclassifications caused by the false constraint relationships. In this paper, we propose a clustering algorithm that combines LLM-generated hard and soft constraints to align with the decision-making characteristics of LLMs.

3 Problem Formulation

In text clustering, a set of n texts T is provided as input, where each short text is transformed into a point in δ-dimensional space $\mathcal{R}^\delta$ by embedding models, resulting in a set of representations (points) S. Let π denote the mapping function from the text $t \in T$ to its embedding point $x = \pi(t) \in S$. For a parameter k, the goal of the text clustering task is to assign these points to k clusters, denoted by $\mathcal{A} = \{A_1, \ldots, A_k\}$, such that the objective function is minimized.

In particular, the traditional k-means algorithm clusters the data based on the means with the goal of minimizing the following objective function:

$$\min \sum_{i=1}^{k} \sum_{x \in A_i \subseteq S} ||x - c(A_i)||^2$$

where cluster A_i represents the collection of points within S and $c(A_i)$ denotes the mass center of A_i. Let C denote the center set involving all the $c(A_i)$ for $i \in [k]$. With the same objective, the constrained k-means needs to additionally satisfy the constraint relationships below.

Let must-link (ML) constraints be defined as $\mathcal{X} = \{X_1, \ldots, X_h\}$, where each $X \in \mathcal{X}$ is an ML set. For each point within the ML set, the clustering considers assigning them to the same center. Similarly, let cannot-link (CL) constraints be defined as $\mathcal{Y} = \{Y_1, \ldots, Y_l\}$, where each $Y \in \mathcal{Y}$ represents a CL set $|Y| \leq k$, and the clustering considers assigning the points in Y to different centers. Note that the hard constraints require the point assignments to strictly follow these constraints, while the soft constraints allow them to violate the constraints by paying a certain penalty.

4 Methodology

In this section, we first propose the algorithmic framework of our approach. Then, we describe the two key stages for constrained text clustering: 1) generating high-accuracy constraint sets while reducing the number of queries; 2) developing a clustering algorithm tailored for utilizing the LLM-generated constraints to improve clustering accuracy.

Algorithmic Framework

In this work, we utilize LLMs to assist in generating both cannot-link and must-link constraints. After embedding text into data points, our method consists of two main stages: 1) constraint generation: we select a set of candidate data points and employ LLMs to assess their relationship, generating ML/CL constraints; 2) constrained clustering: we perform clustering on the data points while incorporating the ML/CL constraint sets generated among the inspected data points. Notably, the selection algorithms used in Stage 1 are tuned to better support the subsequent Stage 2, ultimately yielding clustering results with higher accuracy.

Constraint Generation with LLM

By setting candidate points, we propose a novel LLM-generated constraint algorithm aimed at improving query efficiency while maintaining the quality of constrained clustering. To this end, our candidate query selection algorithm is guided by two key factors: the correctness of the constraints generated by the LLMs and the effectiveness of the queries formulated to assist the LLMs. We provide the key idea of this stage below and further details on the algorithm can be found in the full version.

To enhance query efficiency, we leverage LLMs to generate constraint sets, thereby enhancing the effectiveness of the queries instead of using pairwise constraints (Guo et al. 2024). Specifically, we propose two methods to separately construct candidate constraint sets for the must-links and cannot-links, ensuring that each set contains at least two (≥ 2) points per query. In addition, following previous works (Basu, Banerjee, and Mooney 2004; Mallapragada, Jin, and Jain 2008; Viswanathan et al. 2024), we select candidate points based on distances.

Must-Links Constraint Set. To collect candidate points for the must-link constraints, we employ an algorithm based on coresets (Har-Peled and Mazumdar 2004) to partition the dataset into representative subsets. This approach prevents the selection of points that differ significantly along any single dimension and ensures that each candidate ML set contains mutually similar points. Each sampled subset is mapped to its text and then passed to an LLM, which returns groups of texts. The corresponding points of these text groups are then used to construct the ML sets. In particular, let $X_c = \{x_1, \ldots, x_m\} \subseteq S$ denote a selected must-link candidate point set obtained by the selection method. We map it to the corresponding text set $T_c = \pi^{-1}(X_c) = \{t_1, \ldots, t_m\}$, where π^{-1} is the inverse mapping function from the text to its associated point. The final ML constraint sets are then constructed according to the LLM output as

$$LLM_{ML}(T_c) = \big\{\{t_1, t_2, t_3\}, \ldots, \{t_m\}\big\}.$$

We select each subgroup containing more than one text to form an ML set. For example, given $T = \{t_1, t_2, t_3\}$, we use the mapping π to obtain the corresponding point set $X = \pi(T) = \{x_1, x_2, x_3\}$, which then defines the ML constraint.

Hard and Soft ML Constraints. To effectively leverage LLM-generated constraints in clustering, we assign confidence values to the must-link constraints as indicated by the LLMs' feedback. We separately compute two thresholds for the pairwise and multi-point constraints. The key steps are as follows: 1) According to the coreset algorithm, the points are divided into multiple grid levels r_j, defined as $r_j = (1+\varepsilon)^j \cdot \sqrt{cost_{kc}/10n\delta}$, where $cost_{kc}$ denotes the cost of the k-center problem solved by the min-max algorithm (Gonzalez 1985) and ε is a small constant set to 0.1; 2) At each grid width level, we compute the ordering of the pairwise distances (or set diameters) as $\Psi = \{\max_{x_1,x_2\in X} d(x_1,x_2) \mid X \in S_{r_j}\}$, where S_{r_j} denotes the family of ML sets whose inter-point distances correspond to the jth grid width r_j; 3) We perform a binary search over Ψ to identify a desirable distance threshold $\psi \in \Psi$; 4) For each candidate threshold ψ, we query the LLMs α times ($\alpha = 5$ for pairwise constraints and $\alpha = 10$ for set-based constraints). If all responses are considered consistent, ψ is designated as the maximum allowable diameter for a hard must-link constraint of the corresponding constraint type. In our clustering algorithm, we handle hard and soft ML constraints separately.

Cannot-Links Constraint Set. On the other hand, to collect candidate points for the cannot-link constraints, we uniformly randomly select from the set of uncovered data points whose distance from the current candidate set exceeds a threshold $r = cost_{kc}$, where $cost_{kc}$ is the cost of k-center computed by the min-max algorithm (Gonzalez 1985) to limit the size of the CL constraint set ($\leq k$) regarding its 2-approximation ratio guarantee. This selection strategy is designed to facilitate more accurate judgments by LLMs. To ensure correctness, each sampled point q is evaluated by an LLM to determine whether it should be placed in a cannot-link set. We denote the LLM's response as $LLM_{CL}(T_Y, t_q)$, where Y is the current CL set, $T_Y = \pi(Y)$ is the set of texts mapped from Y via the mapping function π, q is a candidate point and $t_q = \pi(q)$. If $LLM_{CL}(T_Y, t_q)$ returns "None", we append q to Y. Otherwise, we continue to seek the next point. The process continues until the size of Y reaches k, or no new point q can be found beyond the radius r from Y, at which point the algorithm terminates and begins constructing a new CL set.

Clustering with LLM-generated Constraints

Our LLM-generated constraints differ from traditional constraints in two key aspects: 1) they are constructed as sets rather than pairwise, and 2) they may contain erroneous constraints in the LLM's output. In this clustering algorithm, we begin by initializing the clusters using the hard must-link (ML) constraints with the clustering k-means++. For clarity, we next introduce penalty-based constrained clustering algorithms that incorporate CL and ML constraints, respectively. Next, we present a unified framework that can handle both types of constraints. Finally, the algorithm needs to update the centers using the assignment results and repeat the constrained assignment steps until convergence.

Cluster Initialization

Similar to previous studies (Baumann and Hochbaum 2024), we represent hard must-link constraints by their mass center and incorporate them during the seeding step based on the k-means++ initialization strategy (Arthur and Vassilvitskii 2007). The k-Means++ algorithm achieves better clustering accuracy by adding the seeding step to the traditional k-means algorithm, which means the k-means algorithm is sensitive to the improvement of the initial center selection steps. Thus, to keep the advantage, we divide the hard and soft ML constraints during the generation stage and utilize

Algorithm 1: ML-Constrained Clustering with Penalty.

Input: A dataset S with a family of ML sets $\mathcal{X}$ including the hard ML constraint sets $\mathcal{X}_h$.
Output: The assignment result of ML sets $\mathcal{X}$ and center set C.

1 Set $C \leftarrow k$-means++ (Arthur and Vassilvitskii 2007), $P \leftarrow \{P_i\}_{i=1}^{k}$;
`/* Initialization center */`
2 **for** *each ML set* $X \in \mathcal{X}_h$ **do**
3   Set the mass center $\bar{X}$ with $|X|$ to represent the set X;
`/* Assignment step */`
4 **for** *each ML set* $X \in \mathcal{X} \setminus \mathcal{X}_h$ **do**
5   **for** *each point* $x \in X$ **do**
6   Assign x to its nearest center $c_i \in C$;
7   Set partition $P_i \leftarrow P_i \cup x$;
8 **for** *each* $P_i \in P$ *with the largest size* $|P_i|$ **do**
9   Set the mass center $\bar{P}_i$ with $|P_i|$ to represent the partitionP_i; `// similar to` P_j
10   **for** *each* $P_j \in P \setminus P_i$ *with the largest size* $|P_j|$ **do**
11   Set c_{ij} as the nearest center of the merged mass center of $\overline{P_i \cup P_j}$;
12   **if** $(w_m + d(\bar{P}_j, c_j)) \cdot |P_j| + (w_m + d(\bar{P}_i, c_i)) \cdot |P_i| > \sum_{p \in P_i \cup P_j} d(p, c_{ij})$ **then**
13   Set $P_i \leftarrow P_i \cup P_j$;
14 Return $\mathcal{X}' \leftarrow P$ and C.

the hard ML constraints (with high correctness) to join the initial centers selection step. The detailed algorithm is provided in Steps 1-3 of Alg. 1.

ML Clustering with Penalty

As described in the above subsection, we propose distinguishing high-accuracy constraints as hard constraints to serve the initialization step, while treating the remainder as soft constraints. The hard ML constraints will be assigned following their representative point. The key ideas of our method for handling soft constraints are summarized below, with the full algorithm detailed in Alg. 1.

For each soft ML set $X \in \mathcal{X} \setminus X_h$, our approach first assigns the points in X to their partitions in P. To be specific, each point is assigned to its nearest center and divides X into subsets $P_i \in P$, which is represented by its centroid $\bar{P}_i$. Next, we iteratively select the two partitions with the largest diameters, denoted as P_i and P_j, and compare the cost of merging them (obtained by assigning the merged group to the nearest center to its mass center) with the cost of keeping the original assignments and the associated penalties. If the merge yields a lower overall cost, we combine P_i and P_j and update the partition set X accordingly. This process continues until no further merges within X can be merged.

Lemma 1. *The running time of Alg. 1 is* $O(nk^2)$.

Proof. Firstly, the initialization step of Alg. 1 takes $O(n)$ time to process the ML sets $\mathcal{X}$, followed by $O(nk^2)$ for k-means++ to obtain the center set. Then, the assignment step takes $O(nk)$ time to enforce the ML constraints on the partitions, and on all partitions, we spend a total of $O(nk^2)$ time comparing the cost of merging versus splitting each pair of partitions. So the total runtime of Alg. 1 is $O(nk^2)$. □

Algorithm 2: Local Search for CL-Constrained Clustering with Penalty.

Input: A dataset S with a family of CL sets $\mathcal{Y}$ and the center set C.
Output: The assignment result $\mathcal{A}$ of CL sets $\mathcal{Y}$.

for *each CL set* $Y \in \mathcal{Y}$ **do**
    while *True* **do**
        Construct the auxiliary bipartite graph $G(C, Y; E)$ regarding the center set C and the CL set Y;
        Set $M \leftarrow$ Computing the maximum-sum matching on graph G;
        for *each edge* $e(y, c) \in M$ **do**
            Compute the maximum-sum matching regarding graph $G' = G(C, Y \setminus y)$ and set the matching as M';
            Set $num_y \leftarrow 1$;
            for *each* $y' \in Y \setminus y$ **do**
                if $M(y') \neq M'(y')$ **then** num_y++ ;
            Set $g_y \leftarrow \sum_{e(y,c) \in M} d(y, c) - \sum_{e(y,c') \in M'} d(y, c') - d(y, c(y))$;
        if $\max_{y \in Y} g_y < num_y \cdot w_{cl}$ **then**
            Assign CL set Y to $\mathcal{A}$ by matching M;
            break;
        else
            Assign y to A_y with its nearest center $c(y)$;
            Set $Y \leftarrow Y \setminus y$;

CL Clustering with Penalty

To efficiently utilize CL constraint sets, we use the maximum-sum matching approach for k-means clustering. To further reduce the impact of the false relationships, we utilize a penalty term to correct the clustering assignments by local search. The key steps are outlined below, with the algorithm detailed in Alg. 2.

Auxiliary Bipartite Graph Given the center set C and each CL set $Y \in \mathcal{Y}$, we construct an auxiliary bipartite graph $G(Y, C; E)$, where the weight of each edge $e \in E$ is the negative of the distance between its endpoints. The formal definition is as follows.

Definition 1. *Given a center set* C *and a CL set* Y*, the auxiliary bipartite graph* $G(Y, C; E)$ *is defined on the vertex set* $Y \cup C$*, and* E *is the set of edges between* Y *and* C.

Minimum Weight Perfect Matching According to the graph G defined by Def. 1, we compute a maximum-sum matching M to identify centers for the points in each CL set

Algorithm 3: LSCK-HC: ML/CL-Constrained Clustering.

Input: A dataset S of size n, a family of CL sets $\mathcal{Y}$ and a family of ML sets $\mathcal{X}$.
Output: A set of clusters $\mathcal{A}$ regarding center set C.

```
1  Set X' and C ← CALL Alg. 1 regarding the ML constraint sets X;
2  for X' ∈ X' do
3      Compute the mass center X̄' with weight |X'| to represent the set X';
4      Set S ← S ∪ X̄' \ X';
5  Set A ← CALL Alg. 2 regarding datasets S with the family of CL constraint sets Y and C;
6  for each unassigned point s ∈ S do
7      Set S' ← s;
8      if s ∈ X ∈ X then Set S' ← X ;
9      Assign S' to its nearest center c(S') ∈ C;
10     Set A_{c(S')} ← A_{c(S')} ∪ S';
11 Return A.
```

$Y \in \mathcal{Y}$. For a given CL set Y, let M denote the matching between the center set C and Y as below.

Definition 2. *Given an auxiliary bipartite graph $G = (C, Y; E)$, the matching $M_G \subseteq E$ is defined as a minimum weight matching if and only if: 1) M_G is a one-sided perfect matching with $|Y| \leq |C|$; 2) $\sum_{e(y,c) \in M_G} d(y, c)$ attains minimum, where $y \in Y$ and $c \in C$.*

Based on the above definition, the algorithm proceeds as follows. For each matching M: 1) Remove the point y with the largest distance, i.e., $y = \arg\max_{(y,c) \in M} d(y, c)$; 2) Let M' be the maximum-sum matching between $Y \setminus \{y\}$ and C; 3) By comparing M and M', define num_y that is the number of points in $Y \setminus \{y\}$ whose center assignments change and g_y is the corresponding decrease in total cost; 4) If $\max_{y \in Y} g_y < num_y \times w_{cl}$ (where w_{cl} is the penalty), the entire CL set will be assigned according to M; Otherwise, let $y = \arg\max_{y \in Y} g_y$, reassign y to its nearest center $c(y)$, update $Y \leftarrow Y \setminus \{y\}$, and repeat steps 1-4 until for the chosen y we have $g_y \leq num_y \times w_{cl}$.

Lemma 2. *The running time of Alg. 2 is $O(k^{\frac{9}{2}} + nk^4)$.*

Proof. In steps 3-4 of the algorithm, each matching requires $O(|E(G)| + |V(G)|^{\frac{3}{2}}) = O(k^{\frac{3}{2}} + k \cdot |Y|)$ time to compute the maximum matching in G using the algorithm by (van den Brand et al. 2020). Similarly, the matching cost $O(k^{\frac{3}{2}} + k \cdot |Y|)$ in steps 6-9. Thus, when we compare these assignment results, if the cost reduction is always greater than the penalty, we spend $O(\sum_{Y \in \mathcal{Y}} (k^{\frac{3}{2}} + k \cdot |Y|)^2 \times k) = O(k^{\frac{9}{2}} + k^4 \cdot n)$ time to obtain all matching results. □

Clustering with Both ML and CL Constraints

By combining Algs. 1 and 2, we present our overall method for CL and ML constrained clustering in Alg. 3. In this algorithm, we first apply Alg. 1 to assign the ML constraint sets. Considering a point that belongs to both CL and ML relationships, it is crucial to ensure that the assignments respect the centers determined by the ML set in Step 4 of Alg. 3 when addressing the CL constraints.

Lemma 3. *The running time of Alg. 3 is $O((k^{\frac{1}{2}} + n) \cdot k^4)$.*

5 Experiments

In this section, we report the proposed clustering method using LLM-generated constraints. We conduct an extensive comparison with four baseline methods under these constraints. Additionally, we evaluate our approach in terms of query accuracy and query time reduction. We then demonstrate the performance of our algorithm (LSCK-HC) across various embeddings and models. Finally, for constraint generation, we analyze the individual contributions of cannot-link and must-link constraints.

Experiment Setup

Datasets. Following the related previous works (Zhang, Wang, and Shang 2023; Viswanathan et al. 2024), we compare the performance of our algorithms with baselines on the text clustering tasks on five datasets: tweet, banking77, clinc (I/D) and GoEmo.

Baselines. We compare the performance with the closest baseline (Viswanathan et al. 2024), called **FSC**, which uses the min-max method (Mallapragada, Jin, and Jain 2008) to select the query nodes and the PCK-means (Basu, Banerjee, and Mooney 2004) clustering algorithm to cluster the data. For the clustering algorithm, we utilize the ***k*-means++** (Arthur and Vassilvitskii 2007) as one of the baselines. In addition, we list the different popular constrained clustering algorithms based on k-means clustering below: 1) **COP** (Wagstaff et al. 2001): It is the first algorithm designed to address the constrained clustering problem. 2) **PCK** (Basu, Banerjee, and Mooney 2004): It is a constrained clustering algorithm suitable for large data sets with sparse, high-dimensional data (Deng et al. 2024). 3) **BH-KM** (Baumann and Hochbaum 2022): A recent clustering algorithm with soft constraints, which uses a mixed-integer programming formulation, struggles to scale and tends to stop when faced with a large number of constraints; 4) **CKM++** (Jia et al. 2023): It provides an algorithm to handle the hard constraint sets instead of the pairwise constraints. In addition, we set our clustering algorithm without any hard constraints, called **LSCK**.

Metrics. According to the previous works (Zhang, Wang, and Shang 2023; Viswanathan et al. 2024; Huang and He 2024; Zhang, Yuan, and Pan 2024), we report the clustering performance by four metrics: accuracy (ACC) calculated after Hungarian alignment (Kuhn 1955), and Normalized Mutual Information (NMI) calculates mutual information between two assignments, Rand Index (RI) and Adjusted Rand Index (ARI) to quantify the similarity between the predicted clusters and the ground-truth labels.

Experimental Settings. The main experimental results evaluate the clustering with the constraints generated by LLMs. We directly apply these clusterings on extracted embeddings from Instructor-large (Su et al. 2023) and E5 (Wang,

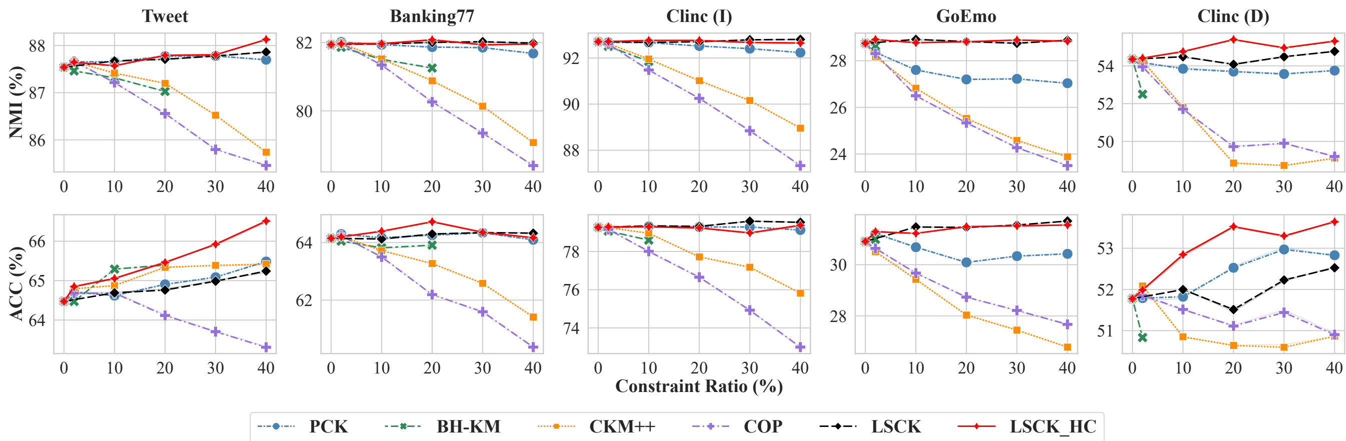

Figure 1: Comparison of average clustering results on datasets across different constrained instance ratios.

Shang, and Zhong 2023) with the same prompt for (Zhang, Wang, and Shang 2023). All experiments were conducted on a Linux machine equipped with an NVIDIA A100 80GB GPU, a 32-core CPU, and 256 GB of RAM, running Python 3.10. Each reported value in this experiment is the average result over 10 runs for each setting.

Clustering Results with Constraints

We report the clustering quality of each algorithm regarding the same constraint sets while varying the ratio of constrained instances. BH-KM does not report all of the constraint ratios due to the difficulty of handling a large number of them.

Combining CL and ML Constraints. In Sec. 4, we provide the query sampling methods for generating cannot-link and must-link constraints, respectively. To evaluate clustering quality, we combine both constraint types through this process: 1) Randomly choose a CL set Y to include in the final constraint collection; 2) If any points in Y are included in a must-link (ML) constraint set, add the relevant ML constraints. Continue repeating steps 1 and 2 until the target constraint ratio is achieved.

In Fig. 1, we show the clustering results for all datasets with constraints alongside the baseline results. The main observations are summarized as follows:

Compared with the baselines, **LSCK-HC and LSCK consistently improve clustering performance over the unconstrained setting.** Across all datasets, our algorithms outperform the 0% constraint clustering results. In particular, clustering accuracy increases by more than 2% as the constraint ratio rises to 40% on the Tweet and CLINC (D) datasets. This improvement surpasses that of the baselines, which we attribute to the effectiveness of local search in finding solutions.

Except for our algorithms, it is observed that PCK generally outperforms COP and CKM++ by including a penalty to handle the constraints as soft constraints in its cost function. However, the effectiveness of this penalty diminishes when the constraint ratio exceeds 10% across multiple datasets (e.g., GoEmo, Clinc (I)), leading to a decline in clustering quality. In contrast, our algorithms leverage a local search method, achieving robust clustering results. **The experimental results are consistent with our design, which can effectively utilize LLM-generated constraints.**

Alg. 3 achieves its best performance with a 20% constraint ratio for most datasets. Moreover, this case can be attributed to two main factors: (1) as the number of constraints increases, the proportion of erroneous constraints also rises; and (2) prior studies including PCK and CKM++ (Basu, Banerjee, and Mooney 2004; Jia et al. 2023) have shown that improvements in clustering quality tend to plateau as more constraints are added, even when all constraints are correct. However, Tweet dataset is an exception in our experiments. We believe this is due not only to the high accuracy of LLM-generated constraints for this dataset, but also to its inherent focus on clustering as the primary task.

Effectiveness and Efficiency of Constraint Generation

In this section, we examine our method described in Sec. 4 with the baseline approach (Viswanathan et al. 2024). We adjust the skeleton structure to better serve the CL constraints and draw inspiration from the coreset technique to develop an ML sampling method. To evaluate constraint effectiveness, we compare both the number of LLM queries and the quality of the resulting constraints by Rand Index.

Query Times. As shown in Tab. 1, we compare our algorithms (see Sec. 4) with the baseline in terms of the number of queries across different constraint ratios and datasets. **Our approach demonstrates a substantial advantage, reducing the number of LLM queries by at least 20-fold for each ratio on all datasets.** In particular, we vary the constraint ratio from 2% to 20% and report the corresponding LLM query counts for CL and ML constraints. We attribute the improvements to the selection method of the candidate constraint sets. Moreover, our proposed selection strategy enables the generation of multiple constrained relationships from a single query.

Constraints Quality. We evaluate the quality of the

Datasets	Methods	2%		10%		20%	
		RI	#Query	RI	#Query	RI	#Query
Bank77-ML	FSC	9.92	5260	8.13	23420	11.40	26580
	Our	**96.42**	**91**	**88.08**	**313**	**85.83**	**480**
Bank77-CL	FSC	99.55	4070	99.32	22620	99.28	37715
	Our	**99.75**	**89**	**99.50**	**641**	**99.48**	**1637**
CLINC-ML	FSC	42.05	25095	69.20	26725	78.97	29225
	Our	**96.25**	**91**	**94.29**	**467**	**91.67**	**927**
CLINC-CL	FSC	**99.57**	12175	99.49	36100	99.48	42135
	Our	99.52	**93**	**99.64**	**662**	**99.59**	**1751**
Tweet-ML	FSC	50.00	13070	74.35	25800	83.82	27425
	Our	**100.00**	**99**	**100.00**	**258**	**99.21**	**539**
Tweet-CL	FSC	**99.74**	5330	99.19	27440	99.17	31485
	Our	99.66	**111**	**99.67**	**323**	**99.56**	**821**

Table 1: Comparison of the LLM-generated constraint quality (RI) and the number of queries (#Query) across different constrained instance ratios.

LLM-generated constraints in Tab. 1, which demonstrates that our method consistently achieves higher accuracy than the FSC. In particular, for must-link (ML) constraints, our algorithm significantly improves correctness. For example, when the constraint ratio is set to 2%, the accuracy of FSC remains below 50%, whereas our method achieves over 96%. This substantial gap is likely due to the baseline deriving ML constraints indirectly, using inference steps that were originally designed for identifying cannot-link (CL) constraints. In contrast, our method explicitly selects the high-confidence ML constraint set based on distance-based criteria. As the constraint ratio increases, the accuracy of FSC also improves, mainly because more ML constraints are generated from steps specifically intended for must-link identification. Nevertheless, our method consistently maintains a higher level of accuracy, outperforming FSC by more than 12% across all settings.

Ablation Studies

As illustrated in Fig. 2 and Tab. 2, we evaluate the algorithms under two different models and embeddings. Then, we measure the influence of the constraints' correctness on the clustering performance, and the separate contributions for CL and ML clustering can be found in the full version.

Robustness for Different Models and Embeddings. In Fig. 2, we compare the performance of two large language models (i.e., DeepSeek R1 and V3) on the Tweet and Banking77 datasets. First, we observe that the quality of the generated constraints varies with the chosen LLM, and those differences in constraint quality directly affect the resulting clustering performance. According to this, we provide the experimental results on how constraint quality affects performance. Next, we find that when evaluating clustering results with the same model, our algorithm outperforms PCK, aligning with the findings discussed in the above section. Thirdly, we show the improvement for our framework (LSCK-HC) over the baseline (FSC) while maintaining a consistent model (V3). For both of the datasets, we discovered that our algorithms exceed baseline performance. We attribute this advantage to the design of our selection strat-

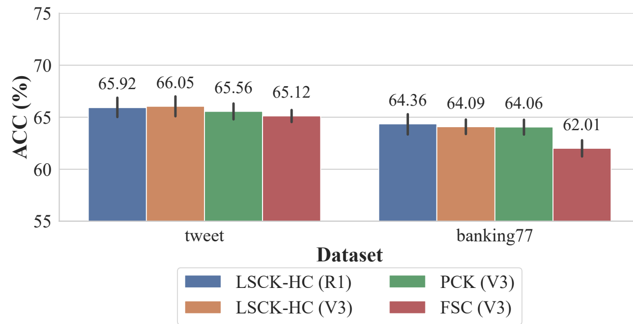

Figure 2: Comparison of clustering accuracy (ACC) across different large language models and algorithms with a 30% constrained-instance ratio. Note that FSC in Viswanathan et al. (2024) uses its own constraint-generation algorithm and applies PCK for clustering.

Embedding	ACC		NMI		ARI	
	PCK	LSCK-HC	PCK	LSCK-HC	PCK	LSCK-HC
Instructor-large	65.08	65.92	87.78	87.80	55.30	55.89
E5	60.18	62.90	85.36	85.86	48.68	58.00

Table 2: Comparison of clustering results on the Tweet dataset at a 30% constrained-instance ratio across different embeddings. Underlined values indicate the best performance for each metric.

egy and clustering algorithm.

As shown in Tab. 2, we compare clustering results for the Instructor-large (Su et al. 2023) and E5 (Wang, Shang, and Zhong 2023) embeddings, using the same language model and evaluated on three metrics. Our algorithm outperforms PCK on both embeddings, and PCK itself yields higher clustering performance with Instructor-large than with E5, which aligns with previous evaluations (Zhang, Wang, and Shang 2023). Under constrained clustering, our method delivers the greatest boost on the smaller E5, improving ARI by nearly 10%. We attribute this to the lower baseline clustering quality of the smaller model, which allows the added constraints to have a stronger corrective effect.

6 Conclusion

In this paper, we present a text clustering algorithm using must-link and cannot-link constraints. Leveraging LLMs to generate constraints, we aim to improve clustering quality through an efficient constraint generation method while reducing resource consumption by treating constraints as sets. We further tune the constrained clustering algorithm to handle these LLM-generated constraints, including both hard and soft ML constraints as well as CL constraints. Moreover, we introduce penalties to mitigate the impact of false constraints and enhance clustering performance by local search. We evaluate our results on five short-text datasets, demonstrating that our method compares favorably in both clustering quality and the cost of constraint generation.

Acknowledgments

This work is supported by National Natural Science Foundation of China (No. 12271098) and Key Project of the Natural Science Foundation of Fujian Province (No. 2025J02011).

References

Ahmed, M. H.; Tiun, S.; Omar, N.; and Sani, N. S. 2022. Short Text Clustering Algorithms, Application and Challenges: A Survey. *Applied Sciences*, 13(1): 342.

Arthur, D.; and Vassilvitskii, S. 2007. k-Means++: The Advantages of Careful Seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1027–1035.

Bae, J.; Helldin, T.; Riveiro, M.; Nowaczyk, S.; Bouguelia, M.-R.; and Falkman, G. 2020. Interactive Clustering: A Comprehensive Review. *ACM Computing Surveys (CSUR)*, 53(1): 1–39.

Basu, S.; Banerjee, A.; and Mooney, R. J. 2002. Semi-Supervised Clustering by Seeding. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*, 27–34.

Basu, S.; Banerjee, A.; and Mooney, R. J. 2004. Active Semi-Supervision for Pairwise Constrained Clustering. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM)*, 333–344. SIAM.

Baumann, P. 2020. A Binary Linear Programming-Based k-Means Algorithm for Clustering With Must-Link and Cannot-Link Constraints. In *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 324–328. IEEE.

Baumann, P.; and Hochbaum, D. 2022. A k-Means Algorithm for Clustering with Soft Must-link and Cannot-link Constraints. In *the 11th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*.

Baumann, P.; and Hochbaum, D. S. 2024. An Algorithm for Clustering With Confidence-Based Must-Link and Cannot-Link Constraints. *INFORMS Journal on Computing*.

Cai, J.; Hao, J.; Yang, H.; Zhao, X.; and Yang, Y. 2023. A Review on Semi-Supervised Clustering. *Information Sciences*, 632: 164–200.

Davidson, I.; and Ravi, S. 2005a. Agglomerative Hierarchical Clustering With Constraints: Theoretical and Empirical Results. In *European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 59–70. Springer.

Davidson, I.; and Ravi, S. 2005b. Clustering With Constraints: Feasibility Issues and the k-Means Algorithm. In *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)*, 138–149. SIAM.

Deng, X.; Liu, J.; Zhong, H.; Feng, F.; Shen, C.; He, X.; Ye, J.; and Wang, Z. 2024. A3S: A General Active Clustering Method With Pairwise Constraints. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 10488–10505.

Feng, X.; Zhang, H.; Zhang, Y.; Zhang, L. Y.; and Pan, S. 2025. BiMark: Unbiased Multilayer Watermarking for Large Language Models. *ArXiv Preprint ArXiv:2506.21602*.

Fu, X.; Xie, W.-B.; Chen, B.; Deng, T.; Zou, T.; and Wang, X. 2024. ACDM: An Effective and Scalable Active Clustering with Pairwise Constraint. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM)*, 643–652.

Ganji, M.; Bailey, J.; and Stuckey, P. J. 2016. Lagrangian Constrained Clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining (SDM)*, 288–296. SIAM.

Gonzalez, T. F. 1985. Clustering To Minimize the Maximum Intercluster Distance. *Theoretical Computer Science*, 38: 293–306.

Guo, L.; Jia, C.; Liao, K.; Lu, Z.; and Xue, M. 2024. Efficient Constrained k-Center Clustering with Background Knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 20709–20717.

Har-Peled, S.; and Mazumdar, S. 2004. On Coresets for k-Means and k-Median Clustering. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing (STOC)*, 291–300.

Huang, C.; and He, G. 2024. Text Clustering As Classification With LLMs. *ArXiv Preprint ArXiv:2410.00927*.

Jia, C.; Guo, L.; Liao, K.; and Lu, Z. 2023. Efficient Algorithm for the k-Means Problem with Must-Link and Cannot-Link Constraints. *Tsinghua Science and Technology*, 28(6): 1050–1062.

Kuhn, H. W. 1955. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1-2): 83–97.

Le, H. M.; Eriksson, A.; Do, T.-T.; and Milford, M. 2018. A Binary Optimization Approach for Constrained k-Means Clustering. In *Asian Conference on Computer Vision (ACCV)*, 383–398. Springer.

Li, Y.; Wang, Y.; Yu, D.-J.; Ye, N.; Hu, P.; and Zhao, R. 2019. Ascent: Active Supervision for Semi-Supervised Learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(5): 868–882.

Lloyd, S. 1982. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2): 129–137.

Mallapragada, P. K.; Jin, R.; and Jain, A. K. 2008. Active Query Selection for Semi-Supervised Clustering. In *ICPR 2008 19th International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society.

Pelleg, D.; and Baras, D. 2007. k-Means With Large and Noisy Constraint Sets. In *European Conference on Machine Learning (ECML)*, 674–682. Springer.

Su, H.; Shi, W.; Kasai, J.; Wang, Y.; Hu, Y.; Ostendorf, M.; Yih, W.-t.; Smith, N. A.; Zettlemoyer, L.; and Yu, T. 2023. One Embedder, Any Task: Instruction-Finetuned Text Embeddings. In *Findings of the Association for Computational Linguistics (ACL)*, 1102–1121.

van den Brand, J.; Lee, Y.-T.; Nanongkai, D.; Peng, R.; Saranurak, T.; Sidford, A.; Song, Z.; and Wang, D. 2020. Bipartite Matching in Nearly-Linear Time on Moderately Dense Graphs. In *Proc. 61st Annu. IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, 919–930. IEEE.

Viswanathan, V.; Gashteovski, K.; Lawrence, C.; Wu, T.; and Neubig, G. 2024. Large Language Models Enable Few-Shot Clustering. *Transactions of the Association for Computational Linguistics (TACL)*, 12: 321–333.

Wagstaff, K.; and Cardie, C. 2000. Clustering with Instance-Level Constraints. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 1103–1110.

Wagstaff, K.; Cardie, C.; Rogers, S.; and Schroedl, S. 2001. Constrained k-Means Clustering with Background Knowledge. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 577–584.

Wang, Z.; Shang, J.; and Zhong, R. 2023. Goal-Driven Explainable Clustering via Language Descriptions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 10626–10649.

Xiong, C.; Johnson, D. M.; and Corso, J. J. 2016. Active Clustering With Model-Based Uncertainty Reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1): 5–17.

Xiong, S.; Azimi, J.; and Fern, X. Z. 2013. Active Learning of Constraints for Semi-Supervised Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 26(1): 43–54.

Yu, Z.; Luo, P.; Liu, J.; Wong, H.-S.; You, J.; Han, G.; and Zhang, J. 2018. Semi-Supervised Ensemble Clustering Based on Selected Constraint Projection. *IEEE Transactions on Knowledge and Data Engineering*, 30(12): 2394–2407.

Zhang, H.; Yu, P. S.; and Zhang, J. 2025. A Systematic Survey of Text Summarization: From Statistical Methods to Large Language Models. *ACM Computing Surveys*, 57(11): 1–41.

Zhang, H.; Yuan, X.; and Pan, S. 2024. Unraveling Privacy Risks of Individual Fairness in Graph Neural Networks. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, 1712–1725. IEEE.

Zhang, Y.; Wang, Z.; and Shang, J. 2023. ClusterLLM: Large Language Models as a Guide for Text Clustering. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 13903–13920.