2/9/2026 4:07:20 PM

Compare Results

Old File:		New File:
2024.emnlp-main.1025.pdf	versus	**2024_emnlp-main_1025.pdf**
8 pages (257 KB)		**8 pages (181 KB)**
10/31/2024 10:33:06 PM		2/9/2026 2:13:56 PM

Total Changes

904

Content

44 Replacements

148 Insertions

379 Deletions

Styling and Annotations

204 Styling

129 Annotations

Go to First Change (page 2)

LLMEdgeRefine: Enhancing Text Clustering with LLM-Based Boundary Point Refinement

Zijin Feng*, Luyang Lin*, Lingzhi Wang†, Hong Cheng, Kam-Fai Wong
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
{zjfeng, lylin, lzwang, hcheng, kfwong}@se.cuhk.edu.hk
*Luyang Lin and Zijin Feng contributed equally.
†Lingzhi Wang is the corresponding author.

February 9, 2026

Abstract

Text clustering is a fundamental task in natural language processing with numerous applications. However, traditional clustering methods often struggle with domain-specific fine-tuning and the presence of outliers. To address these challenges, we introduce **LLMEdgeRefine**, an iterative clustering method enhanced by large language models (LLMs), focusing on edge points refinement. LLMEdgeRefine enhances current clustering methods by creating super-points to mitigate outliers and iteratively refining clusters using LLMs for improved semantic coherence. Our method demonstrates superior performance across multiple datasets, outperforming state-of-the-art techniques, and offering robustness, adaptability, and cost-efficiency for diverse text clustering applications.

1 Introduction

Text clustering is a critical task in various NLP applications, such as topic modeling and information retrieval. Effective clustering enables better data management and more insightful analysis. However, text clustering presents several challenges, particularly in handling edge points—data points that are difficult to assign to clusters due to their ambiguous or extreme characteristics.

The advent of large language models (LLMs) offers new solutions to these challenges. LLMs possess powerful text understanding capabilities that can significantly improve clustering accuracy. For instance, IDAS [?] integrates abstractive summarizations from LLMs directly into clustering processes, and ClusterLLM [?] utilizes LLM-predicted sentence relations to guide clustering.

However, previous LLM-enhanced clustering methods often require extensive LLM API queries, lack domain generalization, or are not sufficiently effective. In this work, we focus on leveraging the text understanding and in-context learning capabilities of LLMs to handle the edge points that traditional methods struggle with.

Our proposed LLMEdgeRefine text clustering method consists of a two-stage clustering edge points refinement processing. Initially, we employ K-means to initialize clusters. In the first stage, we identify edge points using a hard threshold and then form super-points to perform efficient hierarchical secondary clustering. This approach enhances cluster quality by effectively mitigating the effects of outliers. The formation of super-points allows for a more granular examination of cluster boundaries, which is particularly beneficial for accurately delineating ambiguous data points. In the second stage, we leverage the advanced text understanding capabilities of LLMs to refine the cluster edges. This involves a soft edge points removal and re-assignment mechanism, where LLMs reassess and reassign edge points based on their semantic context. This step capitalizes on LLMs' ability to comprehend nuanced text relationships, thereby ensuring more accurate and reliable clustering results.

We validate our method through extensive experiments on eight diverse datasets. The results demonstrate that our method consistently outperforms baseline approaches in terms of clustering accuracy. Additionally, our complexity analysis confirms that our method is more efficient than state-of-the-art techniques, making it a practical choice for large-scale applications.

In summary, our contributions are as follows:

- We introduce a novel two-stage clustering method that effectively refines edge points using LLMs, enhancing clustering accuracy.

- Our method reduces the need for domain-specific fine-tuning and minimizes computational expenses, offering a more efficient solution.
- Comprehensive experimental results demonstrate the superiority of our method in terms of both accuracy performance and efficiency.

2 Related Work

Clustering, a cornerstone of unsupervised learning, has seen diverse applications across various data modalities, including text, images, and graphs [?, ?, ?, ?, ?, ?, ?]. Traditional approaches such as K-means [?] and agglomerative clustering [?] initially dominated, operating on vector representations to partition data based on similarity measures like Euclidean distance or cosine similarity [?, ?].

Recent years have witnessed a paradigm shift towards deep clustering, leveraging deep neural networks to enhance clustering. zhou2022comprehensive categorizes deep clustering into multi-stage [?, ?], iterative [?, ?, ?], generative [?], and simultaneous methods [?, ?].

More recent research has also explored LLM-enhanced clustering. wang2023gpt expands clustering applications to interpretability and explanation generation tasks. In unsupervised clustering, IDAS [?] integrates abstractive summarizations from LLMs directly into clustering processes, highlighting the trend towards leveraging advanced NLP models for clustering tasks. A state-of-the-art method, ClusterLLM [?], utilizes LLM-predicted sentence relations to guide clustering. However, ClusterLLM requires extensive LLM queries and domain-specific fine-tuning, limiting efficiency and generalizability.

Semi-supervised approaches, such as viswanathan2024large, require a subset of ground truth labels or expert feedback, whereas our work focuses on unsupervised clustering.

3 Our Framework

3.1 Problem Formulation

Text clustering takes an unlabeled corpus $\mathcal{D} = \{x_i\}_{i=1}^N$ as input, and outputs a clustering assignment $\mathcal{Y} = \{y_i\}_{i=1}^N$ that maps the input texts to cluster indices. Here, x_i represents individual text instances in the corpus, and y_i represents the cluster index assigned to the text x_i. Given a pre-defined number of cluster K, denote by $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ a clustering of corpus $\mathcal{D}$.

3.2 Our Method

K-means clustering determines cluster centroids based on the mean, which is highly sensitive to extreme values. As a result, outliers – data points significantly different from the majority – can drastically affect centroid positions. Our method follows a four-step process to enhance clustering accuracy by mitigating the effects of outliers and leveraging large language models for improved cluster assignments.

3.2.1 Step 1: Cluster Initialization

We initialize clusters using the K-means algorithm, which partitions data points into K clusters, each represented by a centroid. Denote by $\mathcal{Y}^0 = \{y_i^0\}_{i=1}^N$ the initial clustering assignment, where y_i^0 represents the cluster index assigned to the i-th data point x_i. For simplicity, we use x_i to refer to both the individual text instances and its corresponding embedding representation, with the same applies for other notations. The objective function for K-means is to minimize the sum of squared distances between data points and their corresponding cluster centroids:

$$\min_{\mathcal{Y}^0, \{\mu_j\}_{j=1}^K} \sum_{i=1}^{N} \|x_i - \mu_{y_i^0}\|^2, \tag{1}$$

where μ_j is the centroid of cluster C_j.

3.2.2 Step 2: Super-Point Formation and Re-Clustering

K-means, despite its popularity and efficiency, is known to be sensitive to outliers [?]. In contrast, the agglomerative clustering is often regarded as yielding higher clustering quality [?]. To enhance clustering robustness and

Algorithm 1 Super-Point Enhanced Clustering

Require: Clustering $\mathcal{C}^0$, centroid percentage α, number of iteration γ.
Ensure: Refined clustering $\mathcal{C}'$.

```
t ← 1;
while t ≤ γ do
    C^t ← split(C^{t−1}, α);
    C^t ← agglomerativeClustering(C^t);
    t ← t + 1;
end while
return C' ← C^{t−1};
```

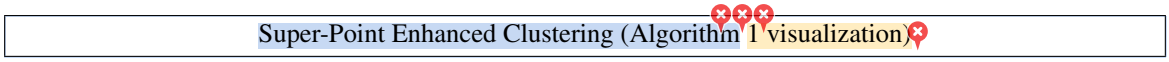

Figure 1: Super-Point Enhanced Clustering

mitigate the impact of outliers, we employ a two-stage process: super-point formation and iterative re-clustering using agglomerative clustering.

[Super-point] Let $\mathcal{C}^t = \{C_1^t, C_2^t, \ldots, C_K^t\}$ be the clustering at iteration t, with μ_j^t as the centroid of cluster C_j^t. For a given percentage α and cluster C_j^t, the super-point S_j^t of C_j^t is defined as the set of the top $\alpha\%$ farthest points from μ_j^t, i.e., $S_j^t = \{x_{i_1}, x_{i_2}, \ldots, x_{i_m} \mid d(x_i, \mu_j^t)$ is among the largest $\alpha\%$ for $x_i \in C_j^t\}$, where $d(x_i, \mu_j^t) = \|x_i - \mu_j^t\|^2$ is the Euclidean distance.

In the super-point formation stage, for each cluster $C_j^t \subset \mathcal{C}^t$, we select the $\alpha\%$ farthest points from the cluster centroid μ_j^t to form super-point S_j^t as defined in Definition 1. The points in S_j^t are aggregated and treated as a single super-point, with the embedding of the super-point being the centroid of S_j^t. This approach allows us to mitigate the effects of outliers by reducing their influence on the overall cluster centroids.

In the re-clustering stage, we start by splitting $\mathcal{C}^t$ into singleton clusters. Each super-point forms its own cluster, i.e., $\{S_j^t \mid j = 1, \ldots, K\}$, while each of the remaining data point is treated as a singleton cluster, i.e., $\{\{x_i\} \mid x_i \in \mathcal{D} \setminus S^t\}$, where $S^t = \cup_{j\in[K]} S_j^t$ is the set of data points in super-points.

Then, we perform the agglomerative clustering to refine the cluster boundaries and enhance intra-cluster homogeneity:

$$\mathcal{Y}^t = \text{Cluster}(\{S_j^t \mid j = 1, \ldots, K\} \cup \{\{x_i\} \mid x_i \in \mathcal{D} \setminus S^t\}) \tag{2}$$

The two-stage process of forming super-points and re-clustering is repeated for γ iterations. By focusing on the central tendencies of clusters while disregarding outliers and noise, this approach improves the overall robustness and quality of the clustering results. The process of Super-Point Enhanced Clustering (SPEC) is depicted in Algorithm **??**. In each iteration of the process, the function `split()` is first called to form super-points and singleton clusters, and then `agglomerativeClustering()` is called to perform re-clustering. In the next step, we leverage LLMs to reassess and reassign the outliers that are far from the re-fined centroids based on their semantic context.

3.2.3 Step 3: Cluster Refinement with Large Language Models

For each reorganized cluster $C_j^t \subset \mathcal{C}^t$, we further refine the clustering by leveraging the contextual understanding of large language models (LLMs). Specifically, we identify the farthest $\beta\%$ of points from the cluster centroid μ_j^t, denoted as V_j. The set of all such points across all clusters is $V = \{V_1, \ldots, V_K\}$. These points are then assessed by LLMs to determine whether they should remain in their current clusters or be reassigned.

Given a clustering $\mathcal{C}$, for each point $x_i \in V$, we query the LLM, denoted as LLMAssessor$(\mathcal{C}, x_i)$, to determine if x_i should be removed from its current cluster. If LLMAssessor$(\mathcal{C}, x_i)$ suggests removal, we reassign x_i to the nearest cluster based on its distance to the centroids:

$$y_i^t = \begin{cases} \arg\min_j \|x_i - \mu_j^t\|, & \text{if removal} \\ y_i^{t-1}, & \text{otherwise} \end{cases} \tag{3}$$

Note that the clustering assignment $\mathcal{Y}$ and clustering $\mathcal{C}$ represent different aspects of clustering and can be deducted from each other. The process will be repeated for l iterations to ensure thorough refinement. The motivation for this step is to utilize the advanced contextual analysis capabilities of LLMs to identify and correct misclassified points, thereby improving the overall clustering accuracy. The algorithm of LLM-Assisted Cluster Refinement (LACR) is illustrated in Algorithm **??**.

Algorithm 2 LLM-Assisted Cluster Refinement

Require: Corpus $\mathcal{D}$, prompt percentage β, number of LACR iterations l, centroid percentage α, number of SPEC iterations γ.
Ensure: clusters $\mathcal{C}$.

$\mathcal{C}^0 \leftarrow \text{KMeans}(\mathcal{D})$;
$\mathcal{C}^1 \leftarrow \text{SecondaryClustering}(\mathcal{C}^0, \alpha, \gamma)$;
$t \leftarrow 1$;
while $t < l$ **do**
  $V' \leftarrow \emptyset$, $V \leftarrow \text{farthestNodes}(\mathcal{C}^t, \beta)$;
  for each $x_i \in V$ **do**
    if $\text{LLMAssessor}(\mathcal{C}, x_i)$ **then**
      $V' \leftarrow V' \cup \{x_i\}$;
    end if
  end for
  $t \leftarrow t + 1$;
  $\mathcal{C}^t \leftarrow \text{re-assign}(\mathcal{C}^{t-1}, V')$;
end while
return $\mathcal{C} \leftarrow \mathcal{C}^t$;

LLM-Assisted Cluster Refinement (Algorithm 2 visualization)

Figure 2: LLM-Assisted Cluster Refinement

Prompting Details. For each data point $x_i \in V$, our method generates a prompt consisting of three main components. Firstly, an instruction inst is crafted to guide the selection process, tailored to the task's context, such as "Select one classification of the banking customer utterances that better corresponds with the query in terms of intent". Secondly, the prompt includes the actual text of the data point x_i itself, forming the core of the query. Finally, our method incorporates a set of eight demonstrations, comprising classification and cluster description pairs. We set the number of demonstrations be eight based on the findings in ??. To simplify the notation, we denote C_k^t as both the k-th nearest cluster to x_i and its description, with the distance measured by the Euclidean distance between the embedding of x_i and the centroid of each cluster. The classification and cluster description pairs are formally defined as $\{(k, C_k^t) \mid k = 1, 2, \ldots, 8\}$. These pairs serve as exemplars to assist in aligning the data point with the appropriate classification.

Remark. Our method focuses on addressing edge data points (outliers) that exhibit extreme characteristics, which are significantly different from the majority of the data. The rationale behind LLMEdgeRefine is to address the limitations of previous clustering methods in handling these edge points and improving cluster cohesion. In Step 1 (§??), K-means provides an initial clustering, but outliers and edge points can distort centroids, resulting in lower clustering quality. Step 2 (§??) introduces super-points to reduce the influence of outliers by focusing on the most representative points in each cluster, enhancing the cluster's internal homogeneity. Step 3 (§??) leverages the contextual understanding of LLMs to further refine the clusters by removing misclassified points, thereby improving the overall clustering accuracy.

In addition to K-means, clustering algorithms that adopt distance metrics and rely on a mean values-based approach also suffer from the impact of outliers. Therefore, our method is portable to these algorithms as well.

4 Experimental Setup

Datasets and Baselines. In our experimental evaluation, we assess LLMEdgeRefine across diverse datasets, including CLINC(I), MTOP(I), Massive(I) [?], GoEmo [?], CLINC-Domain, MTOP-Domain, and Massive-Scenario. These datasets cover intent classification, topic modeling, emotional clustering, and domain-specific scenarios. We compare LLMEdgeRefine against established unsupervised baselines including IDAS [?] and ClusterLLM [?]. The detailed statistics of these datasets is listed in Table ??.

Hyper-Parameters and Experimental Settings. We set parameter K of K-means be the number of ground truth clusters. We adopt modularity [?], a popular metric of the clustering quality without requiring knowledge of the ground truth clustering, as objective function. We automatically determine the values of hyperparameters by

Table 1: Dataset statistics.

Task	Name	#clusters	#data
Intent	CLINC(I)	150	4,500
	MTOP(I)	102	4,386
	Massive(I)	59	2,974
Emotion	GoEmo	27	5,940
Domain	CLINC(D)	10	4,500
	MTOP(D)	11	4,386
	Massive(D)	18	2,974

Table 2: Results (in %) on multiple datasets. Underlines (highlights) indicate top (second) scores per column.

Method	CLINC(I)		MTOP(I)		Massive(I)		GoEmo		CLINC(D)		MTOP(D)		Ma
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	AC
Instructor	79.29	92.60	33.35	70.63	54.08	73.42	25.19	21.54	52.50	56.87	90.56	87.30	61.
SCCL-I	80.85	92.94	34.28	73.52	54.10	73.90	34.33	30.54	54.22	51.08	89.08	84.77	61.
Self-supervise-I	80.82	93.88	34.06	72.50	55.07	72.88	24.11	22.05	58.58	60.84	92.12	88.49	53.
ClusterLLM-I	82.77	93.88	35.84	73.52	59.89	76.96	27.49	24.78	52.39	54.98	93.53	89.36	61.
ClusterLLM	83.80	94.00	35.04	73.83	60.69	77.64	26.75	23.89	51.82	54.81	92.13	89.23	60.
IDAS	81.36	92.35	37.30	72.31	63.01	75.74	30.61	25.57	54.18	63.82	87.57	83.70	53.
LLMEdgeRefine	86.77	94.86	46.00	72.92	63.42	76.66	34.76	29.74	59.40	61.27	92.89	88.19	63.
w/o LACR	85.08	93.71	51.64	73.79	62.21	75.11	25.91	21.19	55.62	57.07	90.57	85.31	60.
w/o LACR & SPEC	77.93	92.31	33.91	71.59	57.17	74.54	34.01	29.31	57.26	56.32	76.85	82.74	59.

conducting a rigorous grid search and select the values that yields the relatively highest modularity score. Besides, our clustering approach utilizes Instructor embeddings [**?**], and for our experiments, we employ the ChatGPT (gpt-3.5-turbo-0301), Llama2 (llama-2-7b-chat), and Mistral (mistral-7B-Instruct-v0.3) as our LLMs.

5 Experimental Results

5.1 Comparison of Effectiveness

We compare the accuracy (ACC) and normalized mutual information (NMI) scores of our method with baselines, and report the results in Table **??**. Table **?.?** demonstrates the effectiveness of LLMEdgeRefine method across multiple datasets. LLMEdgeRefine consistently achieves superior accuracy (ACC) and normalized mutual information (NMI). The method's ability to handle edge points is evident from the significant performance improvements. Specifically, LLMEdgeRefine achieves an average ACC improvement of 17.2%, 10.9%, 17.3%, 11.6%, 12.6%, and 11.1% over Instructor, SCCL-I, Self-supervise-I, ClusterLLM-I, ClusterLLM, and IDAS, respectively, averaging across all tested datasets. In terms of NMI, LLMEdgeRefine outperforms the baselines by an average of 8.4%, 3.8%, 5.4%, 4.3%, 4.8%, and 4.3%, respectively. The ablation study underscores the critical role of LLM-based Adaptive Cluster Refinement (LACR) and Semantic Point Edge Clustering (SPEC) modules, with performance notably dropping when these are removed.

We conduct an ablation study to quantify the impact of various LLMs on effectiveness of our method, and report the results in Table **??**. Table **??** shows that our LLMEdgeRefine on open-sourced LLMs Llama2 and Mistral also demonstrates promising results. This indicates that our method does not purely rely on the powerful text understanding capabilities of close-sourced LLM GPT3.5, highlighting its effectiveness across different LLMs.

5.2 Comparison of Efficiency

The efficiency of our LLMEdgeRefine method is highlighted by its significantly reduced query complexity compared to other models like ClusterLLM [**?**] and IDAS [**?**]. ClusterLLM requires a fixed number of 1618 prompts for each dataset and additional fine-tuning efforts, while IDAS scales with the dataset size, requiring $O(N + |\mathcal{C}|)$ prompts where N is the number of documents and $|\mathcal{C}|$ is the number of clusters. In contrast, LLMEdgeRefine op-

Table 3: Ablation study on clustering quality with various LLMs.

Method	CLINC(I)		MTOP(I)		Massive(I)		GoEmo		CLINC(D)		MTOP(D)	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
LLMEdgeRefine-GPT3.5	86.77	94.86	46.00	72.92	63.42	76.66	34.76	29.74	59.40	61.27	92.89	88.19
LLMEdgeRefine-Llama2	86.60	94.72	46.04	72.93	62.90	76.31	34.50	29.55	59.26	60.93	92.54	87.78
LLMEdgeRefine-Mistral	86.69	94.81	45.88	72.91	63.18	76.48	34.47	29.56	59.48	61.74	92.64	87.84

erates with $O(N \times \beta \times l)$ prompts, where β is a small fraction of N and l is the number of iterations. The detailed complexity analysis can be found in Appendix. For our experiments, with $\beta = 0.1$ and $l = 3$, LLMEdgeRefine demonstrates superior efficiency, reducing the number of prompts needed and thereby improving computational performance without compromising clustering quality.

5.3 Discussion of Hyper-Parameters

We determine the hyper-parameters (i.e., β and l) used in the LACR module based on the results of Bank77 [?] dataset. The sensitivity analysis shows that the clustering quality of our method is not sensitive to the value of β. Specifically, when β varies from 0.1 to 0.9 with a step size of 0.1, the standard deviation of accuracy scores is 0.32 only, indicating stability. For better efficiency, a small β value is sufficient to achieve satisfied performance. The discussion of more hyper-parameters can be found in Appendix.

6 Conclusion

In this work, we introduced LLMEdgeRefine, a novel text clustering method enhanced by LLMs. Our method effectively addresses the challenges posed by outlier data points and domain-specific fine-tuning requirements observed in traditional clustering approaches. The experimental results demonstrate not only the effectiveness but also the efficiency of LLMEdgeRefine.

Limitations While LLMEdgeRefine demonstrates significant improvements in text clustering, several limitations should be noted. Firstly, the method's performance relies on the quality and capacity of the underlying LLMs, which can vary depending on the dataset and domain specificity. Secondly, LLMEdgeRefine requires hyper-parameter tuning, such as the threshold for identifying edge points and the number of iterations, which may not always generalize well across different datasets.

Acknowledgments This work is partially supported by grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14217622).

A Experimental Setup Details

Datasets The statistics of the used datasets are shown in Table ??.

Baselines Apart from SOTA method ClusterLLM and IDAS, we compare other baselines listed in [?].

Hyper-Parameter Selection In Section 5.3, we discussed the selection of β for LLMEdgeRefine. Additionally, we performed a sensitivity test on the Bank77 dataset to determine the optimal number of iterations l for LLM-Assisted Cluster Refinement (LACR), ultimately setting $l = 3$ due to stable performance observed after three iterations. For the hyper-parameters α and γ used in Super-Point Enhanced Clustering (SPEC), we conducted a dataset-specific sensitivity analysis to optimize performance across different datasets. Specifically, we determine the values of hyperparameters by conducting a rigorous grid search and select the values that yields the relatively highest modularity score. This approach allows us to tailor the hyper-parameters to the unique characteristics of each dataset, leading to more accurate and meaningful clustering results. Details of the hyper-parameter selection process are summarized in Tables ?? and ??.

Table 4: Sensitivity test on α, α varies from 0.1 to 0.6 measured by accuracy (ACC) and modularity (MOD).

Method	CLINC(I)		MTOP(I)		Massive(I)		GoEmo		CLINC(D)		MTOP(D)		Massive(S)
	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	Selected α
$\alpha = 0.1$	85.1	91.4	48.1	72.5	63.0	77.0	24.9	46.5	54.4	75.8	90.2	83.0	60.7 78.0
$\alpha = 0.2$	83.4	90.7	47.1	72.3	62.5	77.6	27.9	43.5	47.6	69.9	89.8	82.6	62.7 77.2
$\alpha = 0.3$	82.4	90.0	49.0	72.2	61.1	77.1	27.4	40.7	50.7	72.6	89.1	82.0	60.9 76.8
$\alpha = 0.4$	81.0	89.7	51.7	73.7	63.1	77.8	31.3	42.4	44.1	67.0	88.2	81.4	58.2 74.9
$\alpha = 0.5$	80.1	89.2	51.6	73.7	61.2	77.3	30.3	37.6	40.4	64.3	85.4	81.6	57.5 75.8
$\alpha = 0.6$	80.1	89.4	51.6	73.7	61.2	77.3	30.3	37.6	40.4	64.3	85.4	81.6	57.5 75.8
Selected α	0.1		0.6		0.3		0.1		0.1		0.1		0.1

Table 5: Accuracy scores for different values of γ from 1 to 13 across various datasets.

Method	CLINC(I)		MTOP(I)		Massive(I)		GoEmo		CLINC(D)		MTOP(D)		Massive(S)
	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	ACC	MOD	Selected γ
$\gamma = 1$	85.08	91.4	48.7	64.6	56.9	70.0	25.9	50.2	55.6	77.0	85.3	80.6	59.0 75.7
$\gamma = 2$	84.8	91.2	48.1	70.6	60.0	74.9	27.0	48.3	49.7	72.3	85.4	80.7	57.2 73.4
$\gamma = 3$	85.2	91.1	45.3	71.1	60.1	76.1	25.0	45.4	49.7	69.9	84.7	79.9	59.7 76.6
$\gamma = 4$	85.3	91.2	47.8	72.3	61.8	76.4	24.6	42.9	50.6	69.1	87.6	81.7	59.5 77.8
$\gamma = 5$	85.3	91.2	49.9	73.1	61.0	76.2	25.0	42.7	52.0	74.3	86.5	81.1	60.1 78.0
$\gamma = 6$	85.2	91.2	51.1	73.5	60.9	76.2	24.2	40.4	52.4	72.0	86.3	81.1	58.8 76.6
$\gamma = 7$	84.9	91.1	51.6	73.7	61.2	76.2	23.5	39.9	52.1	72.9	90.6	83.8	60.9 78.5
$\gamma = 8$	84.6	91.1	51.6	73.7	60.4	76.7	26.1	40.5	47.3	70.1	90.5	83.7	60.7 78.2
$\gamma = 9$	84.9	90.9	51.6	73.7	60.4	76.7	26.3	41.8	47.2	71.8	90.7	83.8	60.8 78.2
$\gamma = 10$	84.8	91.0	51.6	73.7	60.4	76.7	26.8	41.1	50.7	75.1	90.6	83.7	60.7 78.2
$\gamma = 11$	84.6	90.8	51.6	73.7	61.1	77.0	27.5	40.7	50.9	74.9	90.6	83.7	60.5 77.7
$\gamma = 12$	84.6	90.8	51.6	73.7	61.1	77.0	27.7	41.4	48.9	74.0	90.6	83.7	59.8 77.4
$\gamma = 13$	84.7	90.8	51.6	73.7	61.1	77.0	27.0	40.0	49.0	74.2	90.1	83.4	60.0 76.8
Selected γ	1		7		5		1		1		7		5

B Complexity Comparison

Complexity of ClusterLLM. Given a set of unlabeled corpus $\mathcal{D}$, in the fine-tuning stage, ClusterLLM constructs 1024 triplet questions and prompts the LLMs with each triplet. In the clustering granularity determination stage, ClusterLLM constructs 594 data pairs by sampling from two clusters that are merged at each step of agglomerative clustering, then prompts the LLMs with each query. In total, ClusterLLM takes 1618 prompts, regardless of the dataset.

Complexity of IDAS. Given a set of unlabeled corpus $\mathcal{D} = \{x_i\}_{i=1}^N$, in the label generation step, IDAS first prompt the LLMs to generate a description of each of the $|\mathcal{C}|$ clusters. Then, for each corpus in $\mathcal{D}$, IDAS constructs and prompts the LLMs. In total, IDAS takes $O(N + |\mathcal{C}|)$ prompts.

Complexity of LLMEdgeRefine. Given a set of unlabeled corpus $\mathcal{D} = \{x_i\}_{i=1}^N$ and a parameter β, at each iteration, our LACR algorithm constructs $N \times \beta$ queries and prompts the LLMs with each query, taking $O(N \times \beta)$ prompts. Over l iterations, our LACR takes $O(N \times \beta \times l)$ prompts in total. In our experiments, we set $\beta = 0.1$ and $l = 3$.