D2CS - Documents Graph Clustering using LLM supervision

Anonymous ACL submission

Abstract

Knowledge discovery from large-scale, heterogeneous textual corpora presents a significant challenge. Document clustering offers a practical solution by organizing unstructured texts into coherent groups based on content and thematic similarity. However, clustering does not inherently ensure thematic consistency. Here, we propose a novel framework that constructs a similarity graph over document embeddings and applies iterative graph-based clustering algorithms to partition the corpus into initial clusters. To overcome the limitations of conventional methods in producing semantically consistent clusters, we incorporate iterative feedback from a large language model (LLM) to guide the refinement process. The LLM is used to assess cluster quality and adjust edge weights within the graph, promoting better intra-cluster cohesion and inter-cluster separation. The LLM guidance is based on a set of success Rate metrics that we developed to measure the semantic coherence of clusters. Experimental results on multiple benchmark datasets demonstrate that the iterative process and additional user-supplied a priori edges improve the summaries' consistency and fluency, highlighting the importance of known connections among the documents. The removal of very rare or very frequent sentences has a mixed effect on the quality scores. Our full code is available here: https://github.com/D2CS-sub/D2CS

1 Introduction

Document clustering and summarization are fundamental tasks in natural language processing (NLP) with numerous applications in information retrieval, content organization, and knowledge discovery (Aggarwal and Zhai, 2012; Wibawa et al., 2024). As the volume of digital text continues to grow exponentially, efficient methods for organizing and extracting meaningful insights from large document collections have become increasingly important (Beltagy et al., 2020; Langston and Ashford, 2024). However, text cohorts are often heterogeneous. As such, two parallel tasks should be performed for content extraction: Text clustering and multi-text summarization. The first task aims to combine similar texts into clusters, and the second aims to summarize all the texts within each cluster. Traditional approaches to document clustering typically rely on conventional distance metrics in high-dimensional vector spaces (Karypis et al., 2000; Zhao et al., 2005), often struggling with the semantic complexity of natural language.

For the summarization, abstraction (shortening and condensing) and extractive (extracting the most important sentences) number of methods were proposed (Nenkova and McKeown, 2012). However, these methods often fail to combine documents that deal with different topics. Recent advancements in neural embeddings (Reimers and Gurevych, 2019a; Su et al., 2022; Melamud et al., 2016) and Large Language Models (LLMs)(Brown et al., 2020; Touvron et al., 2023; Raiaan et al., 2024) have opened new possibilities for improving both clustering quality and summary generation. Given the capacity of LLMS, a simple approach for the summarization of heterogeneous cohorts is to first cluster the document and then summarize each cluster. However, clustering methods are not optimized for thematic consistency. Here, we present Document-to-Cluster Summary with LLMs (D2CS), a novel algorithm that leverages the strengths of modern embedding techniques, density-based filtering, energy distance metrics, and community detection algorithms to create semantically coherent document clusters. Our approach uniquely combines clustering and the generative capabilities of LLMs to produce coherent and fluent cluster summaries, which are iteratively refined through a contrastive evaluation framework. The primary contributions of this work are: A) A comprehensive pipeline integrating sentence-level embeddings with kernel density estimation (KDE) for feature selection,

energy distance for document similarity, and the Leiden algorithm for community detection, B)A novel iterative refinement mechanism leveraging LLM-generated summaries and contrastive document pair evaluations to improve cluster coherence, and C) A multi-faceted evaluation framework that assesses the quality of the clusters and the semantic relevance of the generated summaries.

Algorithm 1 D2CS Pipeline

Input: Sentences set for each document T, k nearest neighbors parameter,prior edges E (optional)

Output: Clustered graph of texts G, summary each cluster $\{C_{sum}\}$

if *E exist* **then**

▷ Create a Graph using prior edges

$G \leftarrow$ **prior_edges_graph**(E)

else

▷ Create an Energy Distance matrix between documents

$D_{ij} \leftarrow \Delta(X_i, X_j), \forall T_i \in T$

▷ Create the base graph

$G \leftarrow$ **create_knn_graph**(D,k)

▷ Cluster and summarize iteratively

foreach $iteration \leq \eta$ **do**

$\mathbb{C} \leftarrow$**cluster_graph**(G)

foreach $C \in \mathbb{C}$ **do**

$C_{sum} \leftarrow$ **summarize**(C,$\forall T_i \in C$)

$SR \leftarrow$**evaluate**($\mathbb{C}, G, T$)

$G \leftarrow$**update_graph**(SR, G)

Return $C_{sum} \forall C \in \mathbb{C}$

2 Related Work

Document Clustering Document clustering has evolved significantly over the decades, from traditional approaches using tf-idf and cosine similarity (Salton and Buckley, 1988) to more sophisticated methods incorporating neural representations (Xie et al., 2016). Early clustering algorithms such as k-means (Hartigan and Wong, 1979) and hierarchical clustering (Murtagh and Contreras, 2012) have been foundational but often struggle with high-dimensional text data. Topic models like Latent Dirichlet Allocation (LDA) (Blei et al., 2003) revolutionized document clustering by introducing probabilistic methods to uncover latent themes. More recent approaches have leveraged neural embeddings, with Doc2Vec (Le and Mikolov, 2014) and SBERT (Reimers and Gurevych, 2019a) enabling more semantically meaningful document representations. The integration of transformers (Vaswani et al., 2017) has further enhanced representation learning for clustering tasks (Zhang et al., 2021; Grootendorst, 2022). Recent work by Liu and Liu (2021) and Ozyurt et al. (2022) has introduced contrastive learning frameworks specifically designed for document clustering, while Alshaikh et al. (2020) demonstrated the effectiveness of mixture-of-experts approaches for multilingual document clustering.

Community detection algorithms have gained prominence in document clustering, with methods like Louvain (Blondel et al., 2008) and Leiden (Traag et al., 2019) showing superior performance in identifying natural clusters. These approaches model documents as graphs, where similarities define edge weights, and have demonstrated effectiveness in capturing complex relationships (Newman and Girvan, 2004; Fortunato, 2010). Recent advances by Shi et al. (2021) and He et al. (2021) have significantly improved the scalability and interpretability of community detection for large document collections. Many measures were defined for the quality of the clusters. However, to the best of our knowledge, no clustering algorithm uses the consistency of texts with their summary as a measure.

Text Summarization As mentioned, text summarization techniques fall broadly into extractive and abstractive categories(Allahyari et al., 2017). Extractive methods (Mihalcea and Tarau, 2004; Erkan and Radev, 2004) select important sentences from documents, while abstractive approaches (See et al., 2017; Lewis et al., 2019) generate novel text that captures the essence of the content.

A different yet related approach is topic detection, where each sentence/document is assigned a domain, and then each domain is summarized by itself by combining either documents belonging to the same domain, or all sentences from multiple documents belonging to the same domain(Radev et al., 2004; Wan and Yang, 2008). Notable recent contributions include Xiao et al. (2021), who developed a pretraining approach specifically for multi-document summarization.

The advent of LLMs has transformed summa-

rization capabilities (Brown et al., 2020; Achiam et al., 2023), with few-shot and zero-shot approaches demonstrating remarkable efficacy. Recent work by Suzgun and Kalai (2024) has shown that guided prompting strategies can substantially improve LLM-based summarization quality, while Tang et al. (2023) have explored the use of LLMs as judges for evaluating summarization outputs. However, to the best of our knowledge, there is no feedback approach to improve the clustering based on the summary evaluation.

Distance Metrics and Feature Selection The choice of distance metrics significantly impacts clustering performance. Beyond traditional measures like Euclidean and cosine distance, more sophisticated approaches such as Earth Mover's distance (Rubner et al., 2000) and energy distance (Székely and Rizzo, 2013) have shown promise for comparing distributions of embeddings. Recent innovations include the optimal transport-based metrics proposed by Huynh and Phung (2021), which have demonstrated superior performance in comparing embedding distributions and contrastive distance learning approaches. We have used an energy distance, which we found to produce the most consistent clusters.

Evaluation of Document Clusters and Summaries Evaluating document clustering remains challenging, with both internal measures like silhouette coefficients (Rousseeuw, 1987) and external measures like normalized mutual information (Strehl and Ghosh, 2002) providing complementary perspectives. For summarization, automatic metrics such as ROUGE (Lin, 2004) and BERTScore (Zhang et al., 2019) quantify lexical and semantic overlap, respectively. More recent evaluation metrics such as QAEval (Deutsch et al., 2021) and SummaC (Laban et al., 2022) have focused on factual consistency and faithfulness of summaries. The integration of human evaluation (Daume III, 2006) and LLM-based assessment (Zhu et al., 2023; Fu et al., 2023) has gained traction for evaluating both clusters and summaries.

Contrastive evaluation methods (Wang et al., 2021) have emerged as a promising approach for refining both clustering and summarization results. Recent work(Bahak et al., 2023) has shown that LLM-based evaluations can achieve high correlation with human judgments across multiple summarization dimensions. Additionally, Luo et al. (2023) and Kocmi and Federmann (2023) have demonstrated that LLMs can effectively serve as judges for evaluating and ranking summaries through pairwise comparisons.

3 Novelty of D2CS

D2CS introduces several novel contributions to the fields of document clustering and summarization:

Integration of Energy Distance with Density-Based Filtering While energy distance(Székely and Rizzo, 2013) has been applied in various domains, its combination with KDE-based filtering for document similarity is unprecedented. Previous approaches Lee et al. (2004) have explored energy distance for clustering but did not address the challenges of high-dimensional embedding spaces or incorporate density-based filtering mechanisms. Our approach efficiently handles the curse of dimensionality by applying KDE to filter out embedding vectors that are either too common or too rare, focusing computational resources on the most informative features. This improves the text quality scores in most cases, but may harm if many documents are extremely rare and hence are completely filtered out by the KDE filter.

Unlike traditional approaches that compute document similarity based on aggregate embeddings (Reimers and Gurevych, 2019a), our method preserves the distribution of sentence-level embeddings within each document. This preserves important structural information about semantic variance within documents.

LLM-Guided Iterative Cluster Refinement The iterative refinement mechanism in D2CS represents a departure from conventional one-pass clustering approaches (Aggarwal and Zhai, 2012). By leveraging LLM-generated summaries as cluster descriptors and using contrastive document pair evaluations to update edge weights, we establish a feedback loop that progressively improves cluster coherence. Previous work (Angelidis et al., 2021) explored LLMs for summarization of predefined clusters, but did not use these summaries to iteratively refine the clusters themselves. Our approach is conceptually related to human-in-the-loop clustering (Cohn et al., 2003) but replaces human feedback with automated LLM assessments, enabling scalability while maintaining quality.

Multi-objective Evaluation Framework The D2CS evaluation framework uniquely combines

technical cluster quality metrics with semantic assessments of summary-document alignment. While prior work has evaluated clusters (Rousseeuw, 1987) and summaries (Lin, 2004) separately, our holistic approach considers both aspects simultaneously. Most notably, our use of contrastive evaluation between in-cluster and out-of-cluster documents relative to the summary breaks new ground in assessing cluster coherence. This methodology builds upon ideas from contrastive learning (Chen et al., 2020) and applies them to the evaluation of document clusters, providing a more nuanced perspective than traditional internal validation metrics.

4 Methods

D2CS leverages an energy distance metric applied to document representations to construct a graph. The weights of the graph's edges are iteratively refined using supervision from a large language model (LLM), with the goal of enhancing cluster coherence and forming homogeneous subgroups of semantically similar documents.

4.1 Graph Generation

To examine our method, we multiple datasets of texts: A) a subset from newsgroups dataset[1], B) content graphs from wikipedia generated using a crawler[2], allowing us to analyze the results on graphs of different sizes and degree distribution., C) a subset from reuters dataset[3], D) Posts that scraped from a given Whatsapp group, focused on middle east political issues [4]. The data or links to the data are accessible on the project Github.

4.2 Document representation via sentence embedding

First, we use a pretrained transformer to compute a representation vector for each sentence in each document $X_i^{(u)} \in \mathbb{R}^d$, where u denotes a single sentence in the given document i sentence set T_i ($u \in T_i$), and d is the dimensionality of the embedding vector.

In some cases, we encountered very long documents, where the large volume of text made it challenging to generate meaningful representations for all sentences. To address this issue, we applied a prompt-based approach using **Cohere**'s LLM to generate a concise summary for each long document.

We used a density-based method (KDE) in order to estimate how common each sentence in the corpus is with a Gaussian kernel density estimation with a radius of 0.075, and filtered out the embeddings within the highest and lowest 7.5% densities.

4.3 Document Graph

Given set of n documents, we construct a distance matrix $D \in \mathbb{R}^{n \times n}$ that represents the pairwise energy distances between documents i and j, where:
$\mathrm{D}_{ij} = \Delta(X_i, X_j) =$
$2 * \frac{1}{|T_i| \cdot |T_j|} \sum_{u \in T_i} \sum_{h \in T_j} ||X_i^{(u)} - X_j^{(h)}||$
$- \frac{1}{|T_i|^2} \sum_{u \in T_i} \sum_{h \in T_i} ||X_i^{(u)} - X_i^{(h)}||$
$- \frac{1}{|T_j|^2} \sum_{u \in T_j} \sum_{h \in T_j} ||X_j^{(u)} - X_j^{(h)}||$, where $||\cdot||$ is the Euclidean distance function.

Using the distance matrix D, we construct a graph G, where vertex V_i represents document i. Edges in G are formed between V_i and its k nearest neighbors (the degree can be above k since we include any edge in a KNN), as determined by D. To ensure efficacy, we build the graph using a BallTree data structure(Omohundro, 1989).

Each edge between e_{ij} is assigned an initial weight $w_{ij} = 1$. In subsequent steps, we employ the LLM to iteratively update w_{ij} to improve the clustering. If the user has additional information on the relation between documents or keywords connecting documents, an additional type of edge is added that is not affected by the summary quality. In the current analysis, we used edges representing common authorship of the text.

4.4 Quality measures

We used five quality measures (Some existing, some novel) for the quality of the clustering, consistency of the abstract, and the text quality and fluency. Each metric tests either a quality of text measure or our novel "clustering success rate":

1. **Quality of text measures** test how close a generated summary is to a human-made one, given the source documents sentences T_i: A) **Relevancy** the quality of passed information, B) **Coherence** the grammar, order and structure, C) **Consistency** the factual correctness, and D) **Fluency** the textual flow and readability.These measures were argued to be more ac-

[1] https://huggingface.co/datasets/SetFit/20_newsgroups
[2] https://github.com/D2CS-sub/wiki-crawler
[3] https://www.kaggle.com/datasets/nltkdata/reuters
[4] https://github.com/D2CS-sub/D2CS/blob/main/posts_1k.csv

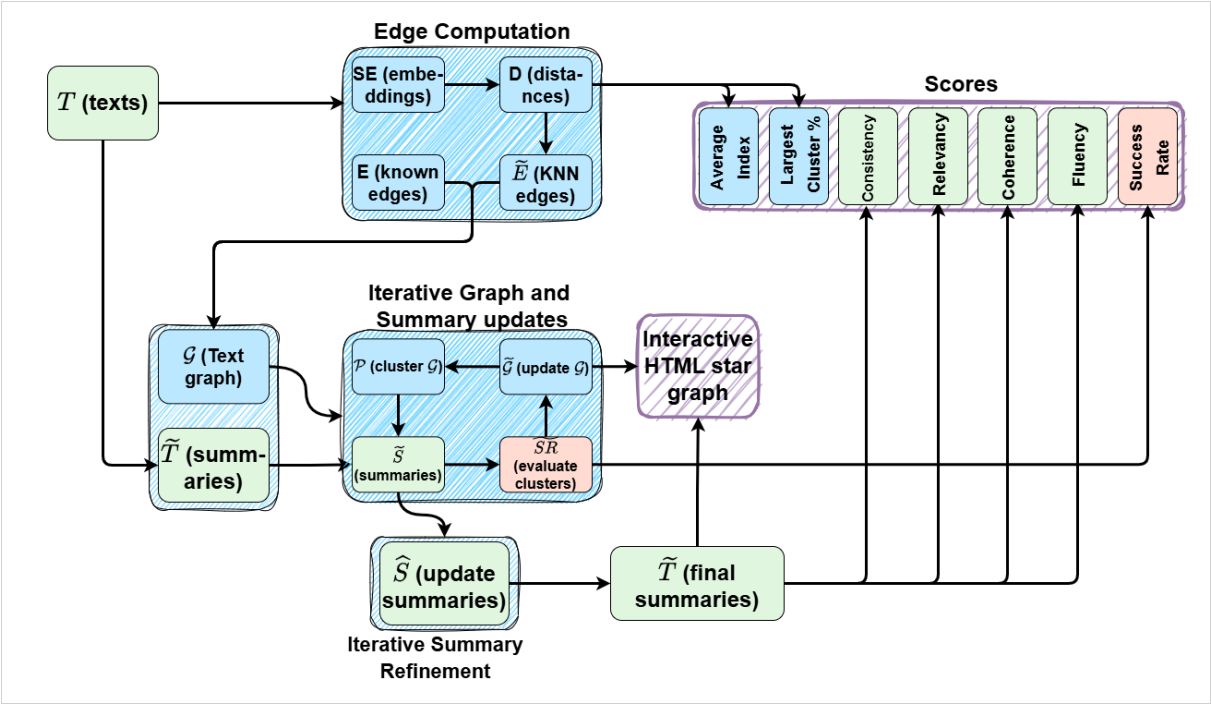

Figure 1: The D2CS pipeline consists of multiple components: iterative clustering, summarizing, and summary refinement. Each sentence in each document is embedded to produce a set of embeddings (SE). The distance between pairs of texts is computed as the energy distance between their embeddings $X_i^{(u)}$. A KNN graph is produced from the distance, and A Leiden algorithm is used to cluster the graph, and each cluster is summarized by itself. The consistency of the texts with the summary of the cluster is used in an iterative process to improve the clustering. The final results are exported to different evaluation scores for either the graph, the summary texts and the coherence between the clusters' documents and their summary.

curate than the more classical ROUGE methods (Fabbri et al., 2021).

2. **Clustering success rate** SR, based on each document SR_i in our graph, defined as:

$$SR = \frac{1}{n}\sum_{i=1}^{n} SR_i = \frac{1}{n}\sum_{i=1}^{n} I_{(A_i \geq B_i)}, \quad (1)$$

where A_i denotes the score assigned by the LLM to indicate how well the text of vertex i fits its assigned cluster summary C_{sum} ($V_i \in C$), and B_i is the score assigned by the LLM to indicate how well text of a randomly selected vertex j fits the same cluster C ($V_j \notin C$, $j \neq i$). Both were estimated by a query to the **Cohere** LLM A.5. The scores A_i, B_i are given to each document, and indicate the validity of the connection between the vertex i and its cluster's summary.

Two additional sanity measures are used on the graph and the clusters to ensure that texts are divided among multiple large clusters, and that the clusters' silhouette is high enough.

4.5 Iterative LLM Guided Graph Clustering

To iteratively improve vertex clustering, we use two LLMs as guides. We apply 4 steps iteratively, until the quality score stop improving.

1. Apply Leiden clustering algorithm (Traag et al., 2019) to the graph to produce clusters.

2. Concatenate the text associated with the documents in each cluster C and generate a cluster summary C_{sum}. The summarization step consists of two LLM-assisted processes:

   (a) The summary process, where the texts themselves are sent along with a specially crafted prompt into **Cohere**, and a drafted summary is received.

   (b) The refinement process, where the summary is sent along with another prompt into **Llama**, and a refined summary is received. The refined summary is crucial for our pipeline since the first step outputs a text with too broad sentences (e.g. "In this text", "It was shown" etc.).

3. Using the generated summaries, compute the clustering success rate (SR_i) for each vertex.

4. Update the graph by increasing the weights of edges connecting vertices from the same cluster if both texts are consistent with the cluster's summary by a factor λ, otherwise decrease the weights of edges the same factor λ (Algorithm 2)

5. After the clustering is done, improve the summaries individually by sending each summary along with its respective cluster's vertices into another summary iteration.

Algorithm 2 Update Document Graph Weights

Input: Graph $G = (V, E)$, vertex scores A_i, B_i for each V_i

Output: Updated edge weights w_{ij} where $(i, j) \in E$

```
foreach C ∈ G do
    foreach (i, j) ∈ E^C do
        if (A_i > B_i) and (A_j > B_j) then
            w_ij ← w_ij * (1 + λ)
        else
            w_ij ← w_ij * (1 − λ)
```

E^C are the edges from E that within cluster C (note that $(V_i, V_j) \in E, V_i \in C$ and $V_j \notin C$) and λ is a scaling factor that determines the magnitude of the increase or decrease in the edge weights. The scores A_i, B_i were given by the **Cohere** LLM as a judge (see here2)

5 Results

Evaluation of D2CS on multiple datasets D2CS clusters documents and then summarizes each cluster. Specifically, D2CS first projects each sentence to an embedding space and defines each text/abstract as a bag of projected sentences (the order is ignored). Then, it creates a graph based on the energy distance between bags and clusters the bags using the Leiden algorithm (Traag et al., 2019). Each cluster is then summarized using an LLM (different LLMs can be chosen). To test that D2CS can produce a coherent summary, we analyzed multiple datasets (as you can see in the appendix Table 2) and tested the performance of D2CS on different sets of abstracts. The evaluation can be performed at multiple levels. The most important measure is the consistency of the clustering. To address that, we used a different LLM and tested whether a cluster summary is more associated with the cluster's abstracts, or random abstracts outside it A.2.

Beyond that 4 measures on the quality of the abstract were proposed in section 4.4. We first tested 8 sets of 100 abstracts each with different keywords (Table 2), and multiple wikipedia tarballs, where we downloaded from each wiki page the first paragraph (See Github for tarball generator). One can see that even this simple vanilla flavor produces very high RS scores, but some of the textual scores are limited (Top sets of bars in Figure 2A and 2B). To test D2CS on more complex datasets, we also analyzed Downloads from News posts (Colors Red and Green in the same plots) that contain more fragmented and heterogeneous text, with similar results. The clusters of D2CS are available in an HTML interface to visualize the clusters, where the edges color represents whether the SR score of this text is high or low (See appendix 3 and the github[5] for the texts used here and the summary of each cluster and a visualization of the clusters).

Iterative clustering improves the scores The vanilla flavor above reaches a high (but not always perfect) RS score, and intermediate scores in the fluency and consistency of text scores. These scores can be iteratively improved by strengthening the edges between texts that are consistent with their common summary and reducing the weights of non-consistent edges. Similarly, the summary can be enhanced by adding refinement queries to improve the text quality. We tested the effect of such queries, and they indeed improve the scores on average(unless they are already close to 1 - Figure 2C). We stopped the iterations when the clusters stopped changing. The iterative process creates a clique of highly consistent documents, and the clustering typically converges after 2 iterations.

When available external information improves performance Given that the goal of the summarization is to produce clusters consistent with the summary, we tested whether additional information associated with the text content can improve the consistency and the quality of the summaries. We tested two types of edges. In the abstracts, we used prior information such as shared keywords, authors, publishing institution etc., and in the Wikipedia pages, we used the edges between the Wikipedia

[5] https://github.com/D2CS-sub/D2CS

texts. The first type of edges is directly associated with shared content, while the Wikipedia edges are associative. As expected, the context-based edges in the $V100$ graphs significantly improved the performance on all scores, while the Wikipedia edges reduced the summaries' scores (Figure 2D).

Filtering rare and frequent sequences is a mixed bag We next tested the effect of density-based filtering. Filtering was applied by computing the non-parametric density, estimated via a KDE algorithm for each sentence. We assumed that sentences in very dense regions are generic, while sentences in very rare regions are too specific. As such, both were removed for the energy distance calculation, but not for the summarization. Specifically, we removed the top and bottom 7.5 % of sentences. The density-based filtering significantly improved the performance over the small dataset (blue bar $p<0.05$), but reduced the accuracy for one of the large datasets (Reuters -green bars). The difference emerges since the larger datasets already had perfect values of RS even without the filtering. As such, we propose filtering as an improvement, only for small sets of texts.

Run time D2CS contains multiple steps: The embedding of each sentence, followed by the distance estimates, clustering, and summarization. While the second and possibly the third stage are proportional to the square of the number of texts or the total number of edges, the last step (the summarization) is linear, but much more computationally extensive than the first steps. As such, in the current applicability ranges (which depend on the total number of tokens allowed by the summarization LLM), the total cost is linear in the number of documents (Appendix Figure 3). Beyond tens of thousands of documents, we expect the cost to be quadratic.

6 Discussion

We have here presented an LLM-guided graph refinement framework that addresses key challenges in document clustering by enhancing semantic coherence through iterative feedback. D2CS combines computational efficiency with the semantic understanding capabilities of LLMs (Kenton and Toutanova, 2019). The combination of clustering and summarizing requires appropriate success rate metrics. We used state-of-the-art metrics and added a clustering coherence metric to ensure, on the one hand, good summarization and, on the other hand, coherence between the summary and the clustering.

While the approach is based on text analysis, we found that user-defined connections significantly guide the clustering process in domain-specific applications. Fully automated methods often struggle to incorporate domain expertise(Chang et al., 2021). This semi-supervised approach aligns with recent work(Wang et al., 2023) showing that minimal human guidance enhances language model performance on complex tasks.

Removing very frequent sentences and very rare sentences improved cluster quality in some types of problems, but not in all. A possible reason for that may be that for specialized corpora, rare sentences contain crucial domain-specific terminology (Reimers and Gurevych, 2019b).

To summarize, D2CS represents a significant advancement in knowledge discovery from heterogeneous textual corpora. The iterative refinement process, guided by LLM feedback and augmented with user-supplied knowledge, effectively overcomes limitations of conventional clustering methods and reaches an approximately 100 % coherence in most datasets. Beyond D2CS, we propose a composite evaluation approach that ensures all aspects of document clustering and summarizing are ensured.

6.1 Limitations

The current approach has multiple limitations. First, clustering quality depends partly on biases present in the LLM, which may propagate to clustering results (Bender et al., 2021). Additionally, the results differ in the components required among domains. We propose a modular approach to adapt to specialized domains with terminology not well-represented in the LLM's training data. Application to very large corpora (tens of thousands of texts) remains challenging due to computational requirements and context limitations of LLM queries. The main limitation is the length of the text to be summarized. This can be addressed by increasing the granularity of clusters or limiting the maximal cluster size. Finally, we have compared different components of the algorithms. However, there are no standard benchmarks for the combined clustering and summarization. Thus, no benchmark comparison was performed. Instead, we propose our current dataset as a benchmark. The datasets are available on the project GitHub.

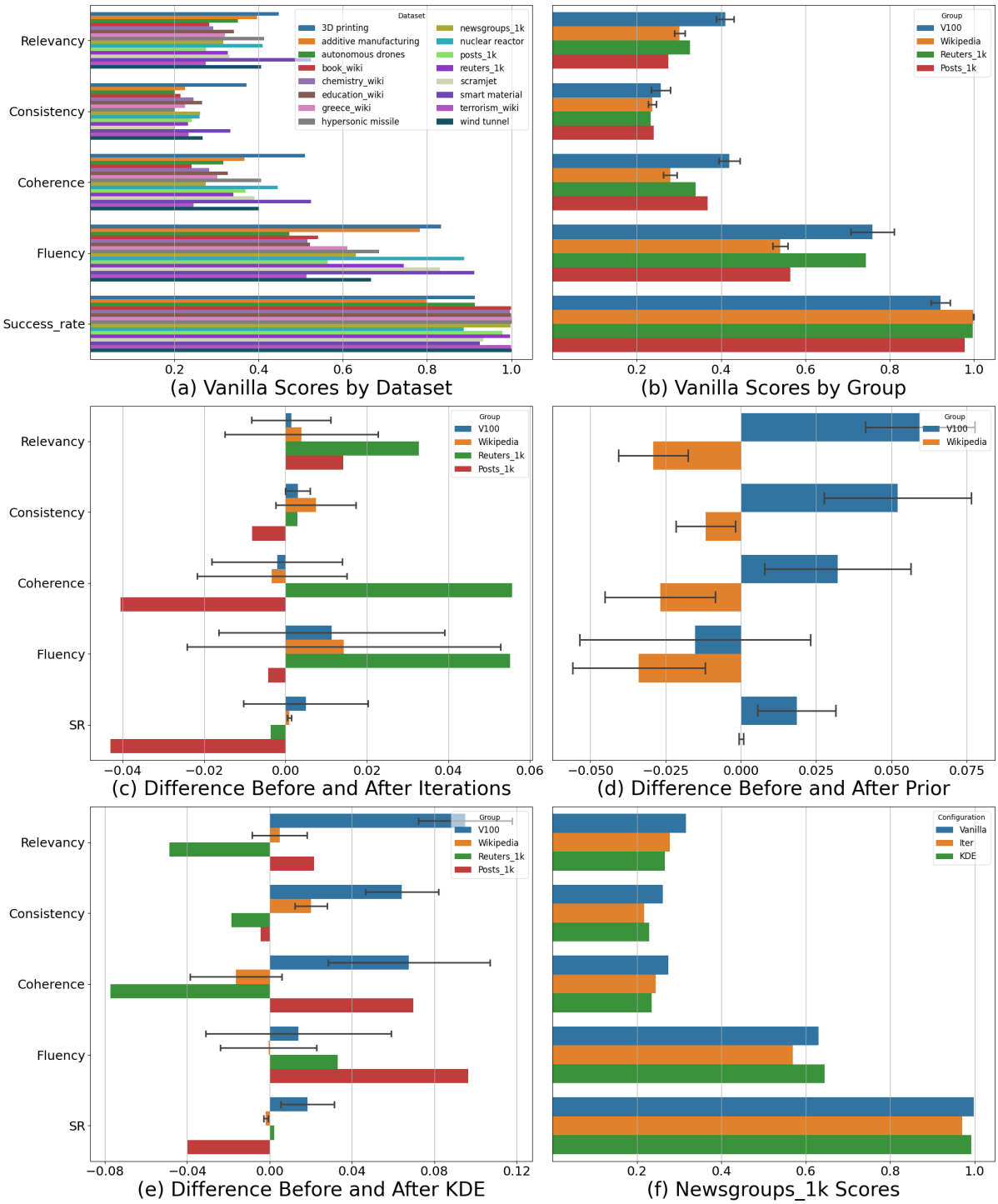

Figure 2: Upper plots, performance of vanilla flavor on all datasets and all quality measures for all datasets (left) and datasets grouped by type (right), where we grouped all wiki datasets, and all small abstract datasets. Second row - effect of adding iterative process (difference from vanilla) and then of adding prior knowledge. Last row - left plot - effect of filtering rare and frequent sentences. right plot - performance on dataset, with no clear effect of any of the additional steps beyond the vanilla flavor.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. *Mining text data*, pages 77–128.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. 2017. Text summarization techniques: a brief survey. *arXiv preprint arXiv:1707.02268*.

Rana Alshaikh, Zied Bouraoui, Shelan Jeawak, and Steven Schockaert. 2020. A mixture-of-experts model for learning multi-facet entity embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5124–5135.

Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Hossein Bahak, Farzaneh Taheri, Zahra Zojaji, and Arefeh Kazemi. 2023. Evaluating chatgpt as a question answering system: A comprehensive analysis and comparison with existing models. *arXiv preprint arXiv:2312.07592*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

David Chang, Eric Lin, Cynthia Brandt, and Richard Andrew Taylor. 2021. Incorporating domain knowledge into language models by using graph convolutional networks for assessing semantic textual similarity: model development and performance comparison. *JMIR medical informatics*, 9(11):e23101.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR.

David Cohn, Rich Caruana, and Andrew McCallum. 2003. Semi-supervised clustering with user feedback. Technical report, Cornell University.

Hal Daume III. 2006. A practical bayesian framework for backpropagation networks. volume 7, pages 448–472. MIT Press.

Daniel Deutsch, Tanya Ro, and Alon Banerjee. 2021. Towards question-answering as an automatic metric for evaluating the content quality of a summary. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 1496–1511.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Santo Fortunato. 2010. Community detection in graphs. *Physics reports*, 486(3-5):75–174.

Jinlan Fu, Abigail See, Sharan Narang, Jungo Jiao, Yizhe Zhang, Sherry Wang, Pengfei Zhong, Dara Bahri, Qin Wang, Chenguang Xiong, and 1 others. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

Chaobo He, Xiang Fei, Qiwei Cheng, Hanchao Li, Zeng Hu, and Yong Tang. 2021. A survey of community detection in complex networks using nonnegative matrix factorization. *IEEE Transactions on Computational Social Systems*, 9(2):440–457.

Viet Huynh and Dinh Phung. 2021. Optimal transport for deep generative models: State of the art and research challenges. In *International Joint Conference on Artificial Intelligence 2021*, pages 4450–4457. Association for the Advancement of Artificial Intelligence (AAAI).

Michael Steinbach George Karypis, Vipin Kumar, and Michael Steinbach. 2000. A comparison of document clustering techniques. In *TextMining Workshop at KDD2000 (May 2000)*, pages 428–439.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1. Minneapolis, Minnesota.

Tom Kocmi and Christian Federmann. 2023. Large language models are state-of-the-art evaluators of translation quality. *arXiv preprint arXiv:2302.14520*.

Philippe Laban, Tobias Hsi, John Canny, and Marti A Hearst. 2022. Summac: Re-visiting nli-based models for inconsistency detection in summarization. In *Transactions of the Association for Computational Linguistics*, volume 10, pages 163–177. MIT Press.

Oliver Langston and Brian Ashford. 2024. Automated summarization of multiple document abstracts and contents using large language models. *Authorea Preprints*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

SangHak Lee, JuneJae Yoo, and TaeChoong Chung. 2004. Distance-based energy efficient clustering for wireless sensor networks. In *29th Annual IEEE International Conference on Local Computer Networks*, pages 567–568. IEEE.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yixin Liu and Pengfei Liu. 2021. Simcls: A simple framework for contrastive learning of abstractive summarization. *arXiv preprint arXiv:2106.01890*.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2023. Chatgpt as a factual inconsistency evaluator for text summarization. *arXiv preprint arXiv:2303.15621*.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 51–61.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Fionn Murtagh and Pedro Contreras. 2012. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. *Mining text data*, pages 43–76.

Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113.

Stephen M Omohundro. 1989. Five balltree construction algorithms.

Yilmazcan Ozyurt, Stefan Feuerriegel, and Ce Zhang. 2022. Contrastive learning for unsupervised domain adaptation of time series. *arXiv preprint arXiv:2206.06243*.

Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

Mohaimenul Azam Khan Raiaan, Md Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. 2024. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE access*, 12:26839–26874.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 2000. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40:99–121.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Jessica Shi, Laxman Dhulipala, David Eisenstat, Jakub Łącki, and Vahab Mirrokni. 2021. Scalable community detection via parallel correlation clustering. *arXiv preprint arXiv:2108.01731*.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. volume 3, pages 583–617.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*.

Mirac Suzgun and Adam Tauman Kalai. 2024. Meta-prompting: Enhancing language models with task-agnostic scaffolding. *arXiv preprint arXiv:2401.12954*.

Gábor J Székely and Maria L Rizzo. 2013. Energy statistics: A class of statistics based on distances. *Journal of statistical planning and inference*, 143(8):1249–1272.

Liyan Tang, Zhaoyi Sun, Betina Idnay, Jordan G Nestor, Ali Soroush, Pierre A Elias, Ziyang Xu, Ying Ding, Greg Durrett, Justin F Rousseau, and 1 others. 2023. Evaluating large language models on medical evidence summarization. *NPJ digital medicine*, 6(1):158.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306.

Dong Wang, Ning Ding, Piji Li, and Hai-Tao Zheng. 2021. Cline: Contrastive learning with semantic negative examples for natural language understanding. *arXiv preprint arXiv:2107.00440*.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Aji Prasetya Wibawa, Fachrul Kurniawan, and 1 others. 2024. A survey of text summarization: Techniques, evaluation and challenges. *Natural Language Processing Journal*, 7:100070.

Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. 2021. Primera: Pyramid-based masked sentence pre-training for multi-document summarization. *arXiv preprint arXiv:2110.08499*.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shangwen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew Arnold, and Bing Xiang. 2021. Supporting clustering with contrastive learning. *arXiv preprint arXiv:2103.12953*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Ying Zhao, George Karypis, and Usama Fayyad. 2005. Hierarchical clustering algorithms for document datasets. *Data mining and knowledge discovery*, 10:141–168.

Jiaxin Zhu, Adam Tauman Kalai, Keziah Zheng, Renrui Zhang, Xue Chen, Wei Zhang, and Jiawei Luo. 2023. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

A Appendix

A.1 Notations

Legend of all the notions we used in this paper and the description of each one of them describe in Table 1.

A.2 Example propmt for summarization

"Instructions: You have received the texts to summarize, separated by '<New Text>' between each consecutive pair of texts. You will summarize the texts according to the following rules: 1- You must not directly reference any of the texts. 2- Your summary must be between 5-10 sentences long. 3- Your summary must mention key ideas and concepts that repeat in the texts. 4- You must not invent any information. The summary must contain only information directly deduced from the texts. 5- Your summary must be coherent, fluent in language, and relevant in content to the texts. 6- In your summary, only refer to the texts you get as input, do not make things up in the summary. 7- Try your best that the summary will be the most relevant, coherent, consistent and as fluent as can be. 8- Do not use characters that are outside the standard ASCII range.

Texts: **[INPUT TEXTS HERE]**"

A.3 Example prompt for refinement

"Instructions:

You will receive an input that contains a summary. Your task is: - Remove any references to the original source (e.g., phrases like "in the article," "this text," or "the provided review") or any wording that refers back to the original document. - After removing these references, rewrite the summary as a single paragraph without line breaks. Maintain the original content without adding any new words or altering the meaning.

Important Rules: - Do not introduce any additional words to the summary. - Ensure the summary remains concise and free of references to the original text. - You will not say anything else. - Do not use characters that are outside the standard ASCII range.

Summary input: **[INPUT SUMMARY HERE]**"

A.4 Example prompt for individual summarization

"In this task you are required to summarize the given text. " "Your summary must be between 5-10 sentences long. " "Your summary must be as coherent as possible, and must not use phrases" "like 'in this text'.Your summary must not include sentences and information " "that is outside the text. may use sentences from the text without citing them."

A.5 Example prompt for SR computation

"Answer using only a number between 1 to 100: " "How consistent is the following summary with the abstract?" "Summary: **[INPUT SUMMARY HERE]**" "Abstract: **[INPUT TEXT HERE]**" "Even if the summary is not consistent with the abstract, please provide a score between " "0 to 100, and only the score," " and only the score, without any '.' or ',' etc."

A.6 Appendix visualization

B Data table

B.1 Run time figure

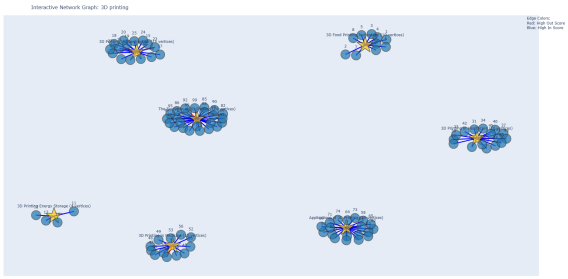

Figure 3: HTML Interface of D2CS results. Each yellow star is an interactive summary, while each blue circle is an interactive abstract from the associated cluster. The edges connecting a vertex i to its summary are blue if $SR_i = 1$, otherwise red.

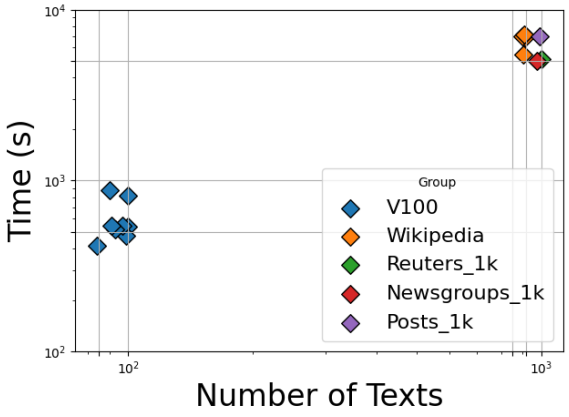

Figure 4: Run time is linear in the number of texts. While the distance requires a comparison between all pairs of text, the summarization is by far the most computationally extensive cost, and is linear in the text length.

Table 1: Legend of notions.

Notion	description
n	Number of documents in the given corpus
i	Document index
T_i	Sentence set of in document i
u	Index of single sentence in document i
d	the dimension of sentence embedding vector
$X_i^{(u)}$	Embedding vector for sentence u from document i ($X_i^{(u)} \in \mathbb{R}^d$)
D	Pairwise distance matrix between documents ($D \in \mathbb{R}^{n \times n}$)
k	The number of nearest neighbors of each document
V	Vertices in the graph. each vertex represent a single document ($\|V\| = n$)
E	The edges between documents vertices. The edges based on nearset neighbors. $(V_i, V_j) \in E$ if document i and document j connected
G	The document undirected Graph ($G = (V, E)$)
η	The Number of iteration for guided clustering process
w_{ij}	The weight of the edge between V_i and V_j
λ	Scaling factor for update edge weight w_{ij} ($\lambda \in [0, 1]$)
$\mathbb{C}$	Set of cluster composed from vertices in G
C	Cluster of vertices in G ($C \in \mathbb{C}$)
E^C	The edges between vertices in the same cluster C. ($(V_i, V_j) \in E^c$ if: $(V_i, V_j) \in E$, $V_i \in C$, $V_j \in C$)
C_{sum}	A summary for all documents in cluster C
A_i	The score given by LLM that V_i belong to the cluster summary C_{sum} ($V_i \in C$)
B_i	The score given by LLM that V_j belong to the cluster summary C_{sum} ($V_j \notin C, V_j \neq V_i, V_i \in C$)
SR_i	Success Rate of V_i to belong to cluser C. boolean parameter. $SR_i = 1$ if $A_i \geq B_i$. Else, $SR_i = 0$
SR	Average for all SR_i scores for each $V_i (V_i \in G)$

Table 2: **Dataset Statistics**: The first column is the dataset name, followed by the Number of vertices in the graph, the number of non-empty texts, the total number of words and of sentences, whether there is user supplied data, and the source. Note that some texts are empty, but are still used for the clustering if there is external data about the text.

Dataset	**N_vertices**	**N_Texts**	**N_Words**	**N_Sentences**	**Prior_Edges**	**Source**
reuters_1k	1000	1000	134047	10290	False	Footnote 3
newsgroups_1k	1000	978	174495	15866	False	Footnote 1
posts_1k	1000	991	27362	2474	False	Footnote 4
terrorism_wiki	1000	903	60159	4041	True	Footnote 2
book_wiki	1000	913	60118	3947	True	Footnote 2
greece_wiki	1000	911	57960	3755	True	Footnote 2
chemistry_wiki	1000	902	59846	4103	True	Footnote 2
education_wiki	1000	913	58670	3986	True	Footnote 2