Large Language Models (LLMs) Enable Few-Shot Clustering

VISWANATHAN Vijay, GASHTEOVSKI Kiril, LAWRENCE Carolin, WU Tongshuang, NEUBIG Graham

Abstract

Unlike traditional unsupervised clustering, semi-supervised clustering allows users to provide meaningful structure to the data, which helps the clustering algorithm to match the user's intent. Existing approaches to semi-supervised clustering require a significant amount of feedback from an expert to improve the clusters. In this paper, we ask whether a large language model (LLM) can amplify an expert's guidance to enable query efficient, few-shot semi-supervised text clustering. We show that LLMs are surprisingly effective at improving clustering. We explore three stages where LLMs can be incorporated into clustering: before clustering (improving input features), during clustering (by providing constraints to the clusterer), and after clustering (using LLMs post-correction). We find incorporating LLMs in the first two stages routinely provides significant improvements in cluster quality, and that LLMs enable a user to make trade-offs between cost and accuracy to produce desired clusters.

Keywords
clustering, few-shot, human-centric

1. Introduction

Unsupervised clustering aims to do an impossible task: organize data in a way that satisfies a domain expert's needs without any specification of what those needs are. Clustering, by its nature, is fundamentally an *underspecified* problem. According to Caruana[1], this under specification makes clustering "probably approximately useless."

Semi-supervised clustering, on the other hand, aims to solve this problem by enabling the domain expert to guide the clustering algorithm[2]. Prior works have introduced different types of interaction between an expert and a clustering algorithm, such as initializing clusters with hand-picked seed points[3], specifying pairwise constraints[4][5], providing feature feedback[6], splitting or merging clusters[7], or locking one cluster and refining the rest[8]. These interfaces have all been shown to give experts control of the final clusters. However, they require significant effort from the expert. For example, in a simulation of split/merge, pairwise constraint, and lock/refine interactions on a toy dataset[8], it took between 20 and 100 feedback interactions to get any clustering algorithm to produce clusters that match a user's specifications. For large, real-world datasets with a large number of possible clusters, the feedback cost required by interactive clustering algorithms can be immense.

Building on a body of recent work that uses LLMs as

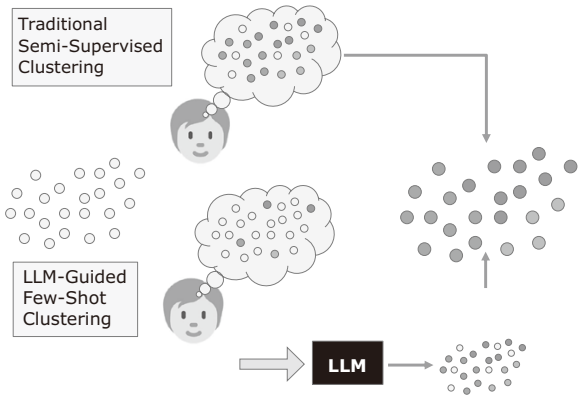


Fig. 1 In traditional semi-supervised clustering, a user provides a large amount of feedback to the clusterer. In our approach, the user prompts an LLM with a small amount of feedback. The LLM then generates a large amount of pseudo-feedback for the clusterer.

noisy simulations of human decision-making[9)-11)], we propose a different approach for semi-supervised text clustering. In particular, we answer the following research question: *Can an expert provide a few demonstrations of their desired interaction (e.g., pairwise constraints) to a large language model, then let the LLM direct the clustering algorithm?* (**Fig. 1**)

We explore three places in the text clustering process where an LLM could be leveraged: before clustering, during clustering, and after clustering. We leverage an LLM *before clustering* by augmenting the textual representation. For each example, we generate keyphrases with an LLM, encode these keyphrases, and add them to the base representation. We incorporate an LLM *during clustering* by adding cluster constraints. Adopting a classical algorithm for semi-supervised clustering, we use an LLM as a pairwise constraint pseudo-oracle. We then explore using an LLM *after clustering* by correcting low-confidence cluster assignments using the pairwise constraint pseudo-oracle. In every case, the interaction between a user and the clustering algorithm is enabled by a prompt written by the user and provided to a large language model.

We test these three methods on five datasets across three tasks: canonicalizing entities, clustering queries by intent, and grouping tweets by topic. We find that, compared to traditional K-Means clustering on document embeddings, using an LLM to enrich each document's representation empirically improves cluster quality on every metric for all datasets we consider. Using an LLM as a pairwise constraint pseudo-oracle can also be highly effective when the LLM is capable of providing pairwise similarity judgements but requires a larger number of LLM queries to be effective. However, LLM post-correction provides limited upside. Importantly, LLMs can also approach the performance of *traditional semi-supervised clustering with a human oracle* at a fraction of the cost.

Our work stands out from recent deep-learning-based text clustering methods[12)13)] in its simplicity. Two of our three methods using an LLM to expand documents' representation or using an LLM to correct clustering outputs can be added as a plug-in to a*ny text clustering algorithm* using *any set of text features.*[*1] In our investigation of what aspect of the LLM prompt is most responsible for the clustering behavior, we find that just using an instruction alone (with no demonstrations) adds significant value. This can motivate future research directions for integrating natural language instructions with a clustering algorithm.

2. Methods to Incorporate LLMs

In section 2, we describe the methods that we use to incorporate LLMs into clustering.

2.1 Clustering via LLM Keyphrase Expansion

Before any cluster is produced, experts typically know what aspects of each document they wish to capture during clustering. Instead of forcing clustering algorithms to mine such key factors from scratch, it could be valuable to globally highlight these aspects (and thereby specify the task emphases) beforehand. To do so, we use an LLM to make every document's textual representation *task-dependent*, by enriching and expanding it with evidence relevant to the clustering need. Specifically, each document is passed through an LLM which generates keyphrases. These keyphrases are encoded by an embedding model, and the keyphrase embedding is then concatenated to the original document embedding.

We generate keyphrases using GPT-3 (specifically, gpt-3.5-turbo-0301). We provide a short prompt to the LLM, starting with an instruction (e.g. "*I am trying to cluster online banking queries based on whether they express the same intent. For each query, generate a comprehensive set of keyphrases that could describe its intent, as a JSON-formatted list.*"). The instruction is followed by four demonstrations of keyphrases, which resemble the example on the upper half of **Fig. 2**.

We then encode the generated keyphrases into a single vector, and concatenate this vector with the original document's text representation. To disentangle the knowledge from an LLM with the benefits of a better encoder, we encode the keyphrases using the same encoder as the original text.[*2]

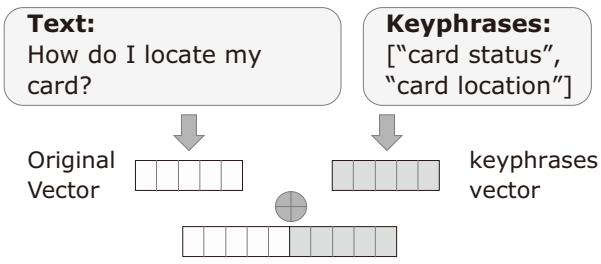


Fig. 2 We expand document representations by concatenating them with keyphrase embeddings. The keyphrases are generated by a large language model.

[*1] On the other hand, pairwise constraint clustering requires using K-Means as the underlying clustering algorithm.
[*2] An exception to this is entity clustering. There, the BERT encoder has been specialized for clustering Wikipedia sentences, so we use DistilBERT to support keyphrase clustering.

This approach is similar to contemporaneous work by Raedt et al.[14], who generate keyphrases for unsupervised intent discovery.

2.2 Pseudo-Oracle Pairwise Constraint Clustering

Arguably, the most popular approach to semi-supervised clustering is *pairwise constraint clustering*, where an oracle (e.g. a domain expert) selects pairs of points which *must* be linked or *cannot* be linked[15], such that the abstract clustering intentions of a user can be implicitly induced from their concrete feedback. In other words, a user conceptually describes which kinds of points to group together and wants to ensure the final clusters follow this grouping. We use this paradigm to investigate the potential of LLMs to amplify expert guidance during clustering by using an LLM as a *pseudo-oracle*.

To select pairs to classify, we take different strategies for entity canonicalization and for other text clustering tasks. For text clustering, we adapt the Explore-Consolidate algorithm[4] to first collect a diverse set of pairs from embedding space (to identify pairs of points that cannot be linked), then collect points that are nearby to already-chosen points (to find pairs of points that must be linked). For entity canonicalization, where there are so many clusters that very few pairs of points must be linked, we simply sample the closest distinct pairs of points in embedding space.

We prompt an LLM with a brief domain-specific instruction, followed by up to 4 demonstrations of pairwise constraints, obtained from test set labels. We use these pairwise constraints to generate clusters with the PCK-Means algorithm[4](**Fig. 3**). This algorithm applies penalties for cluste assignments that violate any constraints, weighted by a hyperparameter w. Following Vashishth et al.[16], we tune this parameter on each dataset's validation split. Due to the potential unreliability of pseudo-oracle pairwise constraints, we initialize our clusters using K-Means++[17] rather than directly using the pairwise constraint neighborhood structure as in prior work[4].

2.3 Using an LLM to Correct a Clustering

We finally consider the setting where one has an existing set of clusters, but wants to improve their quality with minimal local changes. We use the same pairwise constraint pseudo-oracle as in section 2.2 to achieve this, and we illustrate this procedure in **Fig. 4**.

We identify the *low-confidence points* by finding the k points with the least margin between the nearest and second-nearest clusters (setting k = 500 for our experiments). We textually represent each cluster by the entity nearest to the centroid of that cluster in embedding space. For each low-confidence point, we first ask the LLM whether this point is correctly linked to any of the representative points in its currently assigned cluster. If the LLM predicts that this point should not be linked to the current cluster, we consider the 4 next-closest clusters in embedding space as candidates for reranking, sorted by proximity. To rerank the current point, we ask the LLM whether this point should be linked to the representative points in each candidate cluster. If the LLM responds positively, then we reassign the point to this new cluster. If the LLM responds negatively for all alternatives, we maintain the existing cluster assignment.

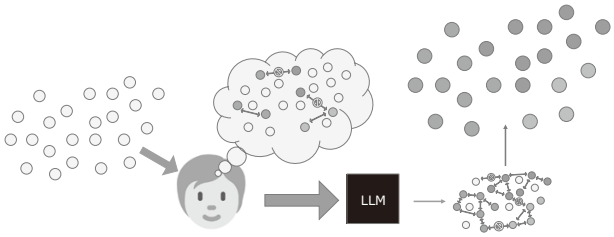
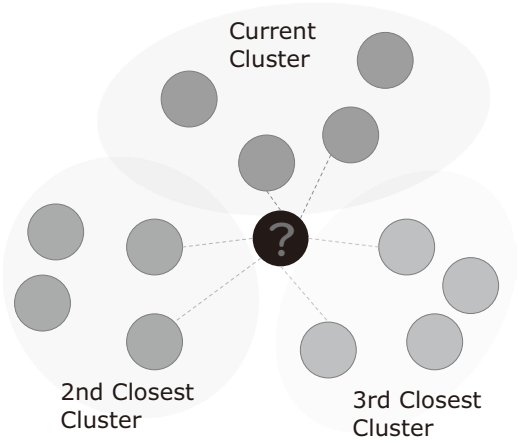


Fig. 3 We use an LLM to generate pairwise constraints for a given dataset, given up to four examples of valid pairwise constraints. The pairwise constraint K-Means ("PCK-Means") algorithm then consumes these "pseudo-oracle" constraints to produce clusters.



After performing clustering, we identify low-confidence points. For these points, we ask an LLM whether the current cluster assignment is correct. If the LLM responds negatively, we ask the LLM whether this point should instead be linked to any of the top-5 nearest clusters and correct the clustering accordingly.

Fig. 4 Steps to fix clustering by using an LLM.

3. Tasks

3.1 Entity Canonicalization

Task: In *entity canonicalization*, we must group a col-

lection of noun phrases $M = \{m_i\}_1^N$ into subgroups $\{C_j\}_1^K$ such that $m_1, m_2 \in C_j$ if and only if m_1 and m_2 refer to the same entity. For example, the noun phrases *President Biden* (m_1), *Joe Biden* (m_2) and *the 46th U.S. President* (m_3) should be clustered in one group (e.g., C_1). The set of noun phrases *M* are usually the nodes of an "open knowledge graph" produced by an OIE system.[*3] Unlike the related task of entity linking[18)19)], we do not assume that any curated knowledge graph, gazetteer, or encyclopedia contains all the entities of interests.

Entity canonicalization is valuable for motivating the challenges of semi-supervised clustering. Here, there are hundreds or thousands of clusters and relatively few points per cluster, making this a difficult clustering task.

Datasets: We experiment with two datasets:

- *OPIEC59k*[20)] contains 22K noun phrases (with 2,138 unique entity surface forms) belonging to 490 ground truth clusters. The noun phrases are extracted by MinIE[21)22)], and the ground truth entity clusters are anchor texts from Wikipedia that link to the same Wikipedia article.
- *ReVerb45k*[16)] contains 15.5K mentions (with 12295 unique entity surface forms) belonging to 6,700 ground truth clusters. The noun phrases are the output of the ReVerb system[23)], and the "ground-truth" entity clusters come from automatically linking entities to the Freebase knowledge graph. We use the version of this dataset from Shen et al.[20)], who manually filtered it to remove labeling errors.

Canonicalization Metrics: We follow the standard metrics used by Shen et al.[20)]:

- *Macro Precision and Recall*
  - Prec: For what fraction of predicted clusters is every element in the same gold cluster?
  - Rec: For what fraction of gold clusters is every element in the same predicted cluster?
- *Micro Precision and Recall*
  - Prec: How many points are in the same gold cluster as the majority of their predicted cluster?
  - Rec: How many points are in the same predicted cluster as the majority of their gold cluster?
- *Pairwise Precision and Recall*
  - Prec: How many pairs of points predicted to be linked are truly linked by a gold cluster?
  - Rec: How many pairs of points linked by a gold cluster are also predicted to be linked?

We finally compute the harmonic mean of each pair to obtain *Macro F1*, *Micro F1*, and *Pairwise F1*.

3.2 Text Clustering

Task: We then consider the case of clustering short textual documents. This clustering task has been extensively studied in the literature[24)].

Datasets: We use three datasets in this setting:

- *Bank77*[25)] contains 3,080 user queries for an online banking assistant from 77 intent categories.
- *CLINC*[26)] contains 4,500 user queries for a task-oriented dialog system from 150 intent categories, after removing "out-of-scope" queries[13)].
- *Tweet*[27)] contains 2,472 tweets from 89 categories.

Metrics: Following prior work[12)], we compare our text clusters to the ground truth using normalized mutual information and accuracy (obtained by finding the best alignment between ground truth and predicted clusters using the Hungarian algorithm[28)]).

4. Baselines

4.1 K-Means on Embeddings

We build our methods on top of a baseline of K-Means clustering[29)] over encoded data with K-Means++ cluster initialization[17)]. We choose the features and number of cluster centers that we use by task, largely following previous work.

Entity Canonicalization: Following prior work[16)20)], we cluster individual entity mentions (e.g. "ever since the ancient Greeks founded the city of *Marseille* in 600 BC.") by representing unique surface forms (e.g. "Marseille") globally, irrespective of their particular mention context. After clustering unique surface forms, we compose this cluster mapping onto the individual mentions (extracted from individual sentences) to obtain mention-level clusters.

We build on the "multi-view clustering" approach[20)]), and represent each noun phrase using textual mentions from the Internet and the "open" knowledge graph extracted from an OIE system. They use a BERT encoder[30)] to represent the textual context where an entity occurs (called the "context view"), and a TransE knowledge graph encoder[31)] to represent nodes in the open knowledge graph (called the "fact view"). They improve these encoders by finetuning the BERT encoder using weak supervision from coreferent entities and improving the knowledge graph representations using data augmentation on the knowledge graph. These two views of each

[*3] Open Information Extraction (OIE) is the task of extracting surface-form (*subject; relation; object*)-triples from natural language text in a schema-free manner.

entity are then combined to produce a representation.

In their original paper, they propose an alternating multiview K-Means procedure where cluster assignments that are computed in one view are used to initialize cluster centroids in the other view. After a certain number of iterations, if the perview clusterings do not agree, they perform a "conflict resolution" procedure to find a final clustering with low inertia in both views. One of our secondary contributions is a simplification of this algorithm. We find that by simply using their finetuned encoders, concatenating the representations from each view, and performing K-Means clustering with K-Means++ initialization[17] in a shared vector space, we can match their reported performance.

Finally, regarding the number of cluster centers, following the Log-Jump method of Shen et al.[13], we choose 490 and 6,687 clusters for OPIEC59k and ReVerb45k, respectively.

Intent Clustering: For the Bank77 and CLINC datasets, we follow[13] and encode each user query using the Instructor encoder. We use a simple prompt to guide the encoder: "Represent utterances for intent classification". Again following previous work, we choose 150 and 77 clusters for CLINC and Bank77, respectively.

Tweet Clustering: Following Zhang et al.[12], we encode each tweet using a version of DistilBERT[32] finetuned for sentence similarity classification[*4, 33]. We use 89 clusters[12].

4.2 Clustering via Contrastive Learning

In addition to the methods described in section 2, we also include two other methods for text clustering, where previously reported: SCCL[12] and ClusterLLM[13]. Both use constrastive learning of deep encoders to improve clusters, making these significantly more complicated and compute-intensive than our proposed methods. SCCL combines deep embedding clustering[34] with unsupervised contrastive learning to learn features from text. ClusterLLM uses LLMs to improve the learned features. After running hierarchical clustering, they also use triplet feedback from the LLM ("is point A more similar to point B or point C?") to decide the cluster granularity from the cluster hierarchy and generate a flat set of clusters. To compare effectively with these approaches, we use the same encoders reported for SCCL and ClusterLLM in prior works: Instructor[35] for Bank77 and CLINC and DistilBERT (finetuned for sentence similarity classification)[32][33] for Tweet.

5. Results

5.1 Summary of Results

We summarize empirical results for entity canonicalization in **Table 1** and text clustering in **Table 2**.[*5] We find that using the LLM to expand textual representations is the most effective, achieving state-of-the-art results on both canonicalization datasets and significantly outperforming a K-Means baseline for all text clustering datasets. Pairwise constraint K-means, when provided

Table 1 Comparing methods for integrating LLMs into entity canonicalization.

Dataset / Method		OPIEC59k				ReVerb45k			
		Macro F1	Micro F1	Pair F1	Avg.	Macro F1	Micro F1	Pair F1	Avg.
Optimal Clustering		80.3	97.0	95.5	90.9	84.8	93.5	92.1	90.1
CMVC		52.8	90.7	84.7	76.1	66.1	87.9	89.4	81.1
K-Means		53.5±0.0	91.0±0.0	85.6±0.0	76.7	69.6±0.0	89.1±0.0	89.3±0.0	82.7
Ours	Keyphrase Clustering	60.3±0.0	92.5±0.0	87.3±0.0	80.0	72.3±0.0	90.2±0.0	90.0±0.0	84.2
Ours	PCKMeans	58.7±0.0	91.5±0.0	86.1±0.0	78.7	72.0±0.0	88.5±0.0	87.0±0.0	82.5
Ours	LLM Correction	58.7	91.5	85.2	78.4	69.9	89.2	88.4	82.5

"CMVC" refers to the multi-view clustering method of Shen et al.[20], while "K-Means" refers to our simplified reimplementation of the same method. Where applicable, standard deviations are obtained by running clustering 5 times with different seeds. Note that the standard deviations being displayed as 0.0 does not mean there was no variance; there was a nonzero standard deviation in most settings but this was less than 0.05 for all experiments.

Table 2 Comparing methods for integrating LLMs into text clustering.

Dataset / Method		Bank77		CLINC		Tweet	
		Acc	NMI	Acc	NMI	Acc	NMI
SCCL		-	-	-	-	78.2	89.2
ClusterLLM		71.2	-	83.8	-	-	-
K-Means		64.0±0.0	81.7±0.0	77.7±0.0	91.5±0.0	57.5±0.0	80.6±0.0
Ours	Keyphrase Clustering	65.3±0.0	82.4±0.0	79.0±0.0	92.6±0.0	62.0±0.0	83.8±0.0
Ours	PCKMeans	59.6±0.0	79.6±0.0	79.6±0.0	92.1±0.0	65.3±0.0	85.1±0.0
Ours	LLM Correction	64.1	81.9	77.8	91.3	59.0	81.5

"SCCL" refers to Zhang et al.[12] while "ClusterLLM" refers to Zhang et al.[13]. We use the same base encoders as those methods in our experiments. Where applicable, standard deviations are obtained by running clustering 5 times with different seeds.

[*4] This model's name is distilbert-base-nli-stsbmean-tokens on HuggingFace.

[*5] As discussed in section 4, when performing entity canonicalization, we assign mentions to the same cluster if they contain the same entity surface form (e.g. "*Marseille*"), following prior work[16][20]. This approach leads to irreducible errors for polysemous noun phrases (e.g. "Marseille" may refer to the athletic club Olympique de Marseille or the city Marseille). To our knowledge, we are the first to highlight the limitations of this "surface form clustering" approach. We present the optimal performance under this assumption in Table 1, finding that the baseline of Shen et al.[20] is already near-optimal on some metrics, particularly for ReVerb45k.

with 20K pairwise constraints pseudo-labeled by an LLM, achieves strong performance on 3 of 5 datasets (beating the current state-of-the-art on OPIEC59k). Below, we conduct more in-depth analyses on what makes each method (in-)effective.

5.2 Illustrative Examples & Key Factors

To qualitatively examine the impact of each LLM-based modification on the clustering process, we use the OPIEC59k dataset to compare the clusters obtained from our various clustering strategies with the clusters obtained from the K-Means baseline.

After aligning each clustering against the ground-truth using the Hungarian algorithm[28], we compute the Jaccard similarity between each predicted cluster and its corresponding ground truth cluster. Comparing the clusters obtained through our LLM-based interventions against the baseline K-Means clusters, we identify clusters where each intervention provides the greatest improvement and the clusters where the intervention causes the greatest degradation.[*6]

While we show one improved cluster and one degraded cluster (relative to the K-Means baseline), these do not occur in equal proportions. In **Table 3**, we show the number of improved and degraded clusters for each method. In **Fig. 5**, **6**, and **7**, we show examples of clusters after *keyphrase expansion, incorporating pairwise constraints,* and *LLM post correction*, and use them to provide intuitions for the key factors affecting each algorithm. On OPIEC59k, it is clear that all our LLM-based interventions mostly lead to improved clusters.

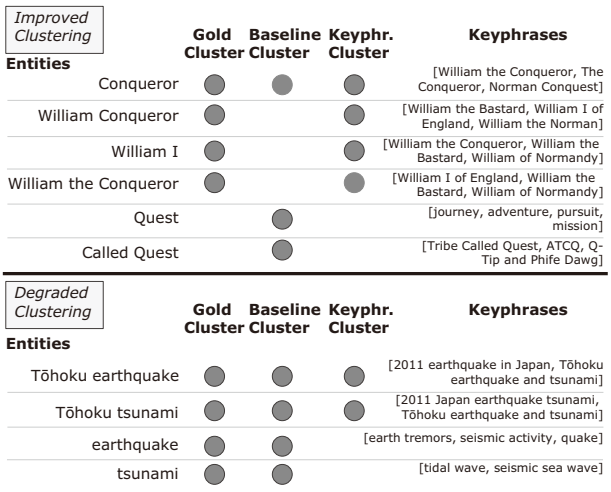
Keyphrase clustering: Providing the right granularity for disambiguation

In Fig.5, we see that LLM-generated keyphrases can disambiguate entities effectively (e.g. generating very different keyphrases for "Conqueror" and "Quest", while the embedding-based baseline clustering incorrectly groups these two). In the degraded example, we also see that these keyphrases may overly focus on each entity's surface form rather than their textual context. This suggests room for more precise modeling and prompt engineering for leveraging keyphrases for complex documents.

Table 3 After aligning the output of each clustering algorithm with the ground truth, we report the number of clusters that were improved or worsened.

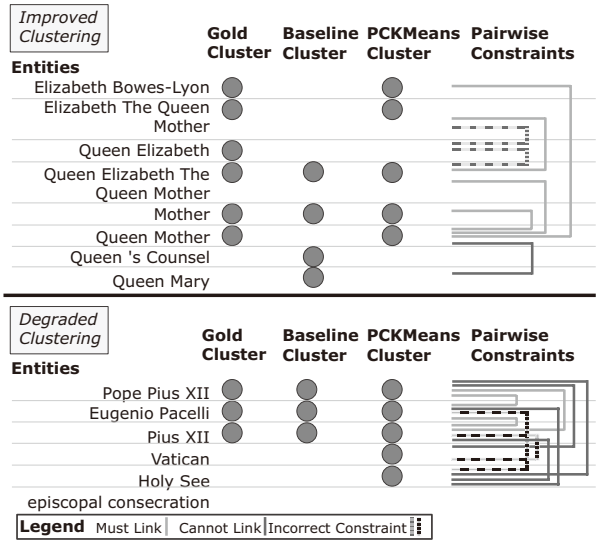
Method	# Improved	# Degraded
K-Means	0	0
Keyphrase Clustering	168	83
PCKMeans	155	82
LLM Correction	102	51

We measured by Jaccard similarity with the corresponding ground truth cluster. Each algorithm produced 490 clusters.



We identify clusters that changed after encoding and clustering keyphrases for each entity. Note that while we provide both the entity name and textual context about the entity to the clusterer, here we omit the textual context for display purposes.

Fig. 5 Example of keyphrases expansion.



We identify clusters that changed after incorporating pairwise constraints and display the relevant pairwise constraints generated by the pseudo-oracle.

Fig. 6 Example of incorporating pairwise constraints.

[*6] We ignore clusters whose output from either algorithm has zero overlap with the corresponding ground truth cluster, since these may be due to cluster misalignment during evaluation.

PCKMeans: Incorrect and conflicting constraints can have too much impact

As shown in Fig. 6, in the improved case, the LLM accurately identifies relationships between some points (e.g. "Mother" and "Queen Mother") which were not grouped together by K-Means clustering on embeddings. In the degraded case, we see a case where the LLM generates conflicting constraints, leading to false positives. While the LLM correctly predicts that "Eugenio Pacelli" and "Pius XII" must be linked and "Pius XII" and "Holy See" cannot be linked, it incorrectly predicts a link between "Eugenio Pacelli" and "Holy See". As a result of these conflicting constraints, the PCKMeans algorithm incorrectly groups additional points into the cluster. **Table 4** provides the accuracy for the pairwise constraints for some datasets, including OPIEC59k.

LLM Correction: Final, hard constraints can lead to over-correction

In the degraded cluster in Fig. 7, we see that the LLM fails to understand the granularity of this cluster, which should focus on The Academy Awards in general rather than a particular award presented at that ceremony. Despite the overall effectiveness of LLM correction for OPIEC59k (Table 3), this example highlights a downside of this approach: we take an absolute decision from the LLM for each point.

This finality impacts the effectiveness of LLM post-correction. In Table 1 and Table 2, the method consistently provides small gains on datasets over all metrics – between 0.1 and 5.2 absolute points of improvement. In Table 4, we see that when we pro- vide the top 500 most-uncertain cluster assignments to the LLM to reconsider, the LLM only reassigns points in a small minority of cases. Though the LLM pairwise oracle is usually accurate, the LLM is disproportionately inaccurate for points where the original clustering already had low confidence.

Improved Clustering

Entities	Gold Cluster	Baseline Cluster	Corrected Cluster	Correction Made?	Prev. Cluster
Charles Grey	●	●	●	No	
Grey	●		●	**Yes**	[film scores, film, film score]
2nd Earl Grey	●	●	●	No	
Earl Grey	●	●	●	No	
William Silent		●		**Yes**	[Charles Grey, Grey, Earl Grey]

Degraded Clustering

Entities	Gold Cluster	Baseline Cluster	Corrected Cluster	Correction Made?	Prev. Cluster
Oscar	●	●	●	No	
Oscars	●	●	●	No	
Academy-Award	●	●	●	No	
Academy Awards	●	●	●	No	
Award for Best Actress			●	**Yes**	[Best Supp. Actor, Supporting Role]
Best Actor			●	**Yes**	[Best Actor in Lead Role, Best Actor]
Award for Best Actor			●	**Yes**	[Best Actor in Lead Role, Best Actor]

We identify clusters that changed after post-correcting cluster assignments with an LLM.

Fig. 7 Example cluster after modification by an LLM.

Table 4 Accuracy of pairwise constraints on several datasets including OPIEC59k.

Datasets / Metrics	OPIEC59k	Tweet	Bank77
Data Size	2138	4500	3080
Total Acc. of Pair. Constraints	86.7	96.8	81.7
# of LLM Reassignmnts	109	78	108
Acc. of Reassignments	55.0	89.7	41.7

When re-ranking the top 500 points in each dataset, the LLM rarely disagrees from the original clustering, and when it does, it is frequently wrong.

5.3 Ablation Study: Why do LLMs Excel at Text Expansion?

In Table 1 and Table 2, we see that *Keyphrase Clustering* is our strongest approach, achieving the best results on 3 of 5 datasets (and giving comparable performance to the next strongest method, *pseudo-oracle PCKMeans*, on the other 2 datasets). This suggests that LLMs are useful for expanding the contents of text to facilitate clustering.

What makes LLMs useful in this capacity? Is it the ability to specify task-specific modeling instructions, the ability to implicitly specify a similarity function via demonstrations, or do LLMs contain knowledge that smaller neural encoders lack?

We answer this question with an ablation study. For OPIEC59k and CLINC, we consider the Keyphrase Clustering technique but omit either the instruction or the demonstration examples from the prompt. For CLINC, we also compare with K-Means clustering on features from the Instructor model, which allows us to specify a short instruction to a small encoder.

Instructions and demonstrations have complementary gains

Empirically, we find that pro- viding either instructions or demonstrations in the prompt to the LLM enables the LLM to improve cluster quality, but that providing both gives the most consistent positive effect. Qualitatively, we observe that providing instructions but omitting demonstrations leads to a larger set of keyphrases with less consistency, while providing demonstrations without any instructions leads to a more focused group of keyphrases that sometimes fail to reflect the desired aspect (e.g. topic vs. intent).

Instruction-finetuned encoders cannot supply enough knowledge

Why is keyphrase clustering using GPT-3.5 in the instruction-only ("with-out demonstrations") setting better than using Instructor (an instruction-finetuned encoder)? The modest scaling curve suggests that scale is not solely responsible: GPT-3.5 likely contains similar or more parameters than GPT-3 (175B), while Su et al.'s Instructor-base/large/XL[35] contain 110M, 335M parameters, and 1.5B parameters, respectively.

Note that we used two types of prompts: While our prompts for GPT-3.5 are very detailed, we used brief prompts for Instructor following their original design (e.g. "Represent utterances for intent classification"), in addition to experimenting with giving the GPT-3.5 prompt to Instructor-XL (the bottom row of **Table 5**). We see that Instructor-XL performs more poorly on the prompt we give to GPT-3.5. We speculate that today's instruction-finetuned encoders are insufficient to support the detailed, task-specific prompts that facilitate few-shot clustering.

5.4 Using an LLM as a pseudo-oracle is cost-effective

We have shown that using an LLM to guide the clustering process can improve cluster quality. However, large language models can be expensive; using a commercial LLM API during clustering imposes additional costs to the clustering process.

In **Table 6**, we summarize the pseudo-labeling cost of collecting LLM feedback using our three approaches. Among our three proposed approaches, pseudo-labeling pairwise constraints using an LLM (where the LLM must classify 20K pairs of points) incurs the greatest LLM API cost. While PCKMeans and LLM Correction both query the LLM the same number of times for each dataset, Keyphrase Correction's cost scales linearly with the size of the dataset, making this infeasible for clustering very large corpora.

Does the improved performance justify this cost? Can we achieve better results at a comparable cost if we employed a human expert to guide the clustering process instead of an LLM? Since pseudo-labeling pairwise constraints requires the greatest API cost in our experiments, we take this approach as a case study. Given a sufficient amount of pseudo-oracle feedback, we see in **Fig. 8** that pairwise constraint K-Means is able to yield an improvement in Macro F1 (suggesting better purity of clusters) without dramatically reducing Pairwise or Micro F1.

Is this cost reasonable? For the $42 spent on the OpenAI API for OPIEC59k (Table 6), one could hire a worker for 3.8 hours of labeling time, assuming an $11-per-hour wage[36]. We observe that an annotator can label roughly 3 pairs per minute. Then, $42 in worker wages would generate <700 human labels at the same cost as 20K GPT-3.5 labels.

Based on the feedback curve in Fig. 8, we see that

Table 5 Comparison of effectiveness of LLM intervention in the absence of demonstration or instructions.

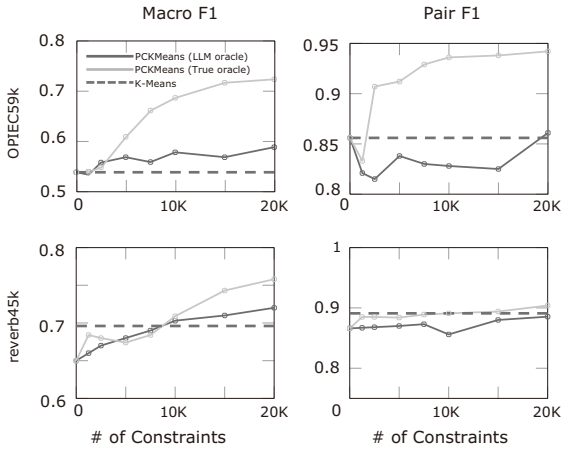
Dataset / Method	OPIEC59k	CLINC	
	Avg. F1	Acc	NMI
Keyphrase Clustering	80.0	79.4 ±0.0	92.6 ±0.0
w/o Instructions	79.1	78.4 ±0.0	92.7 ±0.0
w/o Demonstrations	79.8	78.7 ±0.0	91.8 ±0.0
Instructor-base	- - -	74.8 ±0.0	90.7 ±0.0
Instructor-large	- - -	77.7 ±0.0	91.5 ±0.0
Instructor-XL[35]	- - -	77.2 ±0.0	91.9 ±0.0
Instructor-XL (GPT-3.5 prompt)	- - -	70.8 ±0.0	88.6 ±0.0

We see that GPT-3.5-based Keyphrase Clustering outperforms instruction-finetuned encoders of different sizes, even when we provide the same prompt.

Table 6 Comparison of pseudo-labeling costs of clustering approaches using different LLMs.

Method	Data Size	Cost in USD		
		PCKMeans	Correction	Keyphrase
OPIEC59k	2138	$42.03	$12.73	$2.24
ReVerb45k	12295	$33.81	$10.24	$10.66
Bank77	3080	$10.25	$3.38	$1.23
CLINC	4500	$9.77	$2.80	$0.95
Tweet	2472	$11.28	$3.72	$0.99

We used OpenAI's gpt-3.5-turbo-0301 API in June 2023.



Collecting more pseudo-oracle feedback for pairwise constraint K-Means improves the Macro F1 metric without reducing other metrics. Compared to the same algorithm with true oracle constraints, we see the sensitivity of this algorithm to a noisy oracle.

Fig. 8 Achieving improvement of the Macro F1 of pairwise constraint K-Means.

GPT-3.5 is remarkably more effective than a true oracle pairwise constraint oracle at this price point; unless at least 2,500 pairs labeled by a true oracle are provided, pairwise constraint K-Means fails to deliver any value for entity canonicalization. This suggests that if the goal is maximizing empirical performance, querying an LLM is more cost-effective than employing a human labeler.

6. Conclusion

We find that using LLMs in simple ways can pro- vide consistent improvements to the quality of clusters for a variety of text clustering tasks. We find that LLMs are most consistently useful as a means of enriching document representations, and we believe that our simple proof-of-concept should motivate more elaborate approaches for document expansion via LLMs.

7. Acknowledgements

This work was supported by a fellowship from NEC Laboratories Europe. We are grateful to Wiem Ben Rim, Saujas Vaduguru, and Jill Fain Lehman for their guidance. We also thank Chenyang Zhao for providing valuable feedback on this work.

References

1) Rich Caruana: Clustering: probably approximately useless?, CIKM '13: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp.1259–1260, 2013
https://dl.acm.org/doi/10.1145/2505515.2514692
2) Juhee Bae, Tove Helldin, Maria Riveiro, Sławomir Nowaczyk, Mohamed-Rafik Bouguelia and Göran Falkman: Interactive Clustering: A Comprehensive Review, ACM Computing Surveys, Volume 53 Issue 1, pp.1-39, 2020
https://dl.acm.org/doi/abs/10.1145/3340960
3) Sugato Basu, Arindam Banerjee and Raymond J. Mooney: Semi-supervised Clustering by Seeding, The 19th International Conference on Machine Learning (ICML-2002), pp.19-26, 2002
https://www.cs.utexas.edu/~ml/papers/semi-icml-02.pdf
4) Sugato Basu, Arindam Banerjee and Raymond J. Mooney: Active Semi-Supervision for Pairwise Constrained Clustering, The SIAM International Conference on Data Mining, (SDM-2004), pp.333-344, 2004
https://www.cs.utexas.edu/~ml/papers/semi-sdm-04.pdf
5) Hongjing Zhang, Sugato Basu and Ian Davidson: A Framework for Deep Constrained Clustering - Algorithms and Advances, ECML/PKDD, 2019
https://arxiv.org/abs/1901.10061
6) Sajib Dasgupta and Vincent Ng: Which Clustering Do You Want? Inducing Your Ideal Clustering with Minimal Feedback, Journal of Artificial Intelligence Research, Volume 39, pp.581-632, 2010
https://arxiv.org/abs/1401.5389
7) Pranjal Awasthi, Maria Florina Balcan and Konstantin Voevodski: Local algorithms for interactive clustering, Journal of Machine Learning Research 18, 2017
https://www.jmlr.org/papers/volume18/15-085/15-085.pdf
8) Anni Coden, Marina Danilevsky, Daniel F. Gruhl, Linda Kato and Meena Nagarajan: A method to accelerate human in the loop clustering, SDM 2017, 2017
https://research.ibm.com/publications/a-method-to-accelerate-human-in-the-loop-clustering
9) Jinlan Fu, See-Kiong Ng, Zhengbao Jiang and Pengfei Liu: GPTScore: Evaluate as You Desire, 2023
https://arxiv.org/abs/2302.04166
10) John J. Horton: Large Language Models as Simulated Economic Agents: What Can We Learn from Homo Silicus?, Working Paper 31122, National Bureau of Economic Research, 2023
https://arxiv.org/abs/2301.07543
11) Joon Sung Park, Joseph C. O.'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang and Michael S. Bernstein: Generative Agents: Interactive Simulacra of Human Behavior, 2023
https://arxiv.org/abs/2304.03442
12) Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew O. Arnold and Bing Xiang: Supporting Clustering

with Contrastive Learning, 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5419-5430, 2021
https://arxiv.org/abs/2103.12953
13) Yuwei Zhang, Zihan Wang and Jingbo Shang: ClusterLLM: Large Language Models as a Guide for Text Clustering, 2023
https://arxiv.org/abs/2305.14871
14) Maarten De Raedt, Fréderic Godin, Thomas Demeester and Chris Develder: IDAS: Intent Discovery with Abstractive Summarization, 2023
https://arxiv.org/abs/2305.19783
15) Kiri L. Wagstaff and Claire Cardie: Clustering with Instance-level Constraints, The Seventeenth International Conference on Machine Learning, pp.1103-1110, 2000
https://www.wkiri.com/research/papers/wagstaff-constraints-00.pdf
16) Shikhar Vashishth, Prince Jain and Partha Talukdar: CESI: Canonicalizing Open Knowledge Bases Using Embeddings and Side Information, The 2018 World Wide Web Conference, pp.1317-1327, 2018
https://arxiv.org/abs/1902.00172
17) David Arthur and Sergei Vassilvitskii: k-means++: The Advantages of Careful Seeding, ACM-SIAM Symposium on Discrete Algorithms, 2007
https://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf
18) Razvan C. Bunescu and Marius Pasca: Using Encyclopedic Knowledge for Named Entity Disambiguation, 11th Conference of the European Chapter of the Association for Computational Linguistics, pp.9-16, 2006
https://aclanthology.org/E06-1002/
19) David N. Milne and Ian H. Witten: Learning to link with wikipedia, I CIKM '08: Proceedings of the 17th ACM conference on Information and knowledge management, pp.509-518, 2008
https://dl.acm.org/doi/10.1145/1458082.1458150
20) Wei Shen, Yang Yang and Yinan Liu: Multi-View Clustering for Open Knowledge Base Canonicalization, 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '22, pp.1578-1588, 2022
https://arxiv.org/abs/2206.11130
21) Kiril Gashteovski, Rainer Gemulla and Luciano Del Corro: Minie: Minimizing Facts in Open Information Extraction, 2017 Conference on Empirical Methods in Natural Language Processing, pp.2630-2640, 2017
https://aclanthology.org/D17-1278/
22) Kiril Gashteovski, Sebastian Wanner, Sven Hertling, Samuel Broscheit and Rainer Gemulla, OPIEC: An Open Information Extraction Corpus, Conference on Automatic Knowledge Base Construction (AKBC), 2019
https://arxiv.org/abs/1904.12324
23) Anthony Fader, Stephen Soderland and Oren Etzioni: Identifying relations for open information extraction, EMNLP '11: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp.1535-1545, 2011
https://dl.acm.org/doi/10.5555/2145432.2145596
24) Charu C. Aggarwal and ChengXiang Zhai: A Survey of Text Clustering Algorithms, Mining Text Data, pp.77-128, 2012
https://link.springer.com/chapter/10.1007/978-1-4614-3223-4_4
25) Iñigo Casanueva, Tadas Temcˇinas, Daniela Gerz, Matthew Henderson and Ivan Vulic´ : Efficient Intent Detection with Dual Sentence Encoders, 2nd Workshop on Natural Language Processing for Conversational AI, pp.38–45, 2020
https://aclanthology.org/2020.nlp4convai-1.5/
26) Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang and Jason Mars: An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction, 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp.1311–1316, 2019
https://aclanthology.org/D19-1131/
27) Jianhua Yin and Jianyong Wang: A model- based approach for text clustering with outlier detection, 2016 IEEE 32nd International Conference on Data Engineering (ICDE), pp.625-636, 2016
28) Harold W. Kuhn: The hungarian method for the assignment problem, Naval Research Logistics (NRL), Volume2, Issue1-2, pp.83-97, 1955
https://doi.org/10.1002/nav.3800020109
29) S. Lloyd: Least squares quantization in PCM, IEEE Transactions on Information Theory, Volume 28 Issue 2, pp.129–137, 1982
https://ieeexplore.ieee.org/document/1056489
30) Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp.4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics, 2019
https://arxiv.org/abs/1810.04805
31) Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston and Oksana Yakhnenko: Translating Embeddings for Modeling Multi-relational Data, 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, pp.2787–2795, 2013
https://proceedings.neurips.cc/paper_files/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf
32) Victor Sanh, Lysandre Debut, Julien Chaumond and Thomas Wolf: DistilBERT, a distilled version of BERT:

smaller, faster, cheaper and lighter, 2019
https://arxiv.org/abs/1910.01108
33) Nils Reimers and Iryna Gurevych: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2019
https://arxiv.org/abs/1908.10084
34) Junyuan Xie, Ross B. Girshick and Ali Farhadi: Unsupervised Deep Embedding for Clustering Analysis, International Conference on Machine Learning, 2015
https://arxiv.org/abs/1511.06335
35) Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer and Tao Yu: One Embedder, Any Task: Instruction-Finetuned Text Embeddings, 2022
https://arxiv.org/abs/2212.09741
36) Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch and Jeffrey P. Bigham: A Data-Driven Analysis of Workers' Earnings on Amazon Mechanical Turk, 2018 CHI Conference on Human Factors in Computing Systems, 2017
https://arxiv.org/abs/1712.05796

Authors' Profiles

VISWANATHAN Vijay
Carnegie Mellon University

GASHTEOVSKI Kiril
NEC Laboratories Europe and Center for Advanced Interdisciplinary Research, Ss. Cyril and Methodius Uni. of Skopje

LAWRENCE Carolin
Manager
NEC Laboratories Europe

WU Tongshuang
Carnegie Mellon University

NEUBIG Graham
Carnegie Mellon University

Information about the NEC Technical Journal

Thank you for reading the paper.
If you are interested in the NEC Technical Journal, you can also read other papers on our website.

Link to NEC Technical Journal website

Japanese

English

Vol.17 No.2 Special Issue on Revolutionizing Business Practices with Generative AI
– Advancing the Societal Adoption of AI with the Support of Generative AI Technologies

Remarks for Special Issue on Revolutionizing Business Practices with Generative AI
Approaches to Generative AI Technology: From Foundational Technologies to Application Development and Guideline Creation

Vol.17 No.2
June 2024

Special Issue TOP

Papers for Special Issue

Market Application of Rapidly Spreading Generative AI
NEC Innovation Day 2023: NEC's Generative AI Initiatives
Streamlining Doctors' Work by Assisting with Medical Recording and Documentation
Using Video Recognition AI x LLM to Automate the Creation of Reports
Understanding of Behaviors in Real World through Video Analysis and Generative AI
Automated Generation of Cyber Threat Intelligence
NEC Generative AI Service (NGS) Promoting Internal Use of Generative AI
Utilization of Generative AI for Software and System Development
LLMs and MI Bring Innovation to Material Development Platforms
Disaster Damage Assessment Using LLMs and Image Analysis

Fundamental Technologies that Enhance the Potential of Generative AI
NEC's LLM with Superior Japanese Language Proficiency
NEC's AI Supercomputer: One of the Largest in Japan to Support Generative AI
Towards Safer Large Language Models (LLMs)
Federated Learning Technology that Enables Collaboration While Keeping Data Confidential and its Applicability to LLMs
Large Language Models (LLMs) Enable Few-Shot Clustering
Knowledge-enhanced Prompt Learning for Open-domain Commonsense Reasoning
Foundational Vision-LLM for AI Linkage and Orchestration
Optimizing LLM API usage costs with novel query-aware reduction of relevant enterprise data

For AI Technology to Penetrate Society
Movements in AI Standardization and Rule Making and NEC Initiatives
NEC's Initiatives on AI Governance toward Respecting Human Rights
Case Study of Human Resources Development for AI Risk Management Using RCModel

NEC Information

2023 C&C Prize Ceremony