Clustering Textual Data
Literature Review, PoC, Winter 2024

Salveen Singh Dutt
01166213@pw.edu.pl

Karina Tiurina
01191379@pw.edu.pl

Patryk Prusak
01151475@pw.edu.pl

supervisor: Anna Wróblewska
Warsaw University of Technology
anna.wroblewska1@pw.edu.pl

Abstract

Clustering textual data, such as user-generated comments or reviews, is a critical task in understanding and organizing large datasets. This work explores machine learning techniques to cluster text data effectively, focusing on applications like categorizing Amazon reviews into meaningful groups. By investigating state-of-the-art methods in natural language processing such vectorization and similarity metrics, and leveraging unsupervised learning algorithms, we aim to uncover latent patterns within textual datasets. Our approach aims to demonstrate the potential for automating content categorization, providing actionable insights for e-commerce, social media analysis, and other domains.

1 Introduction

Clustering textual data, such as online reviews, into meaningful categories is a critical yet challenging task, especially in the absence of predefined labels. Reviews can be grouped based on sentiment, topics, or other latent patterns, with no single "correct" clustering. This ambiguity complicates evaluation, as traditional clustering metrics often fail to capture the semantic nuances of text. Furthermore, encoding methods like TF-IDF or BERT embeddings influence how clusters form, adding another layer of complexity. To address these challenges, we propose a pipeline that preprocesses, encodes, clusters, and evaluates text data, with a focus on developing metrics that quantify clustering quality across multiple interpretations. This approach aims to enhance the reliability and interpretability of unsupervised clustering for large-scale textual datasets.

1.1 Research Questions

We aim to explore the following questions:

1. What are the current methodologies for clustering text data, and how effectively do they perform in different contexts?
2. What metrics and evaluation frameworks are used to measure the performance of text clustering, particularly in unsupervised settings
3. What novel approaches or adaptations can enhance the robustness, interpretability, and accuracy of text clustering techniques?

2 State-of-the-Art in Text Clustering and Performance Evaluation

Clustering textual data has advanced significantly with the advent of modern natural language processing (NLP) techniques. These methods predominantly leverage embedding-based representations, deep learning, and hybrid approaches to uncover latent patterns in text data. This section explores notable approaches that have revolutionized clustering methodologies, including embedding models, hybrid frameworks, and specialized clustering techniques.

2.1 Clustering as Classification with LLMs

One prominent method from the paper "Text Clustering as Classification" [6] involves transforming the text clustering problem into a classification task. This involves prompting the LLM to generate potential labels for a given dataset and then assigning the most appropriate label to each sample. This approach not only clusters the data but also provides interpretable labels for each cluster, enhancing the understanding of the underlying structure.

This approach leverages the generative and classification capabilities of LLMs while addressing their context length limitations.

Given an unlabeled dataset $\mathcal{D} = \{d_i\}_{i=1}^{N}$, where N is the corpus size, our goal is to output K subsets of $\mathcal{D}$ as $\mathcal{C} = \{c_j\}_{j=1}^{K}$, where:

- K represents the number of clusters
- Each c_j represents a cluster
- $d_1, d_2 \in c_j$ if d_1 and d_2 belong to the same cluster

The LLM is prompted to generate a set of potential labels for the given dataset by providing examples of data points and asking it to suggest relevant labels. Labels are generated in mini-batches to accommodate input length limitations, followed by merging similar labels to refine cluster granularity This stage generates meaningful cluster labels through the following steps:

1. **Initial Processing:**
   - Input corpus is processed through an embedder (DistilBERT, Instructor, E5)
   - K-means clustering is applied to create initial groupings
   - LLMs are used to generate descriptive labels for these groupings
2. **Label Generation Process:**
   - Dataset is divided into mini-batches of size B (set to 15 in implementation)
   - For each batch $\mathcal{D}'$, LLM generates potential labels: L' = P$_B(\mathcal{T}_{generate}, \mathcal{D}', l)$ where:
     - $\mathcal{T}_{generate}$ is the label generation task instruction
     - l represents n example label names
     - $\mathcal{D}'$ consists of B input data instances
3. **Label Aggregation and Merging:**
   - Unique labels are extracted: L$_{unique}$ $= \{l \mid l \in \mathcal{L}'\}$
   - Similar labels are merged to avoid redundancy: L = P$_m(\mathcal{T}_{merge}, \mathcal{L}_{unique})$ where $\mathcal{T}_{merge}$ represents merging task instructions

Stage 2: Classification Once the labels are generated, the LLM classifies each data point into one of the generated labels. It compares the semantic and contextual similarities of data points with the labels to assign the best-fitting category.
The second stage performs label-based classification:

1. **Label Assignment:**
   - Each document d_j is classified into one of the generated labels: c'$_j$ $= \mathcal{P}_a(\mathcal{T}_{assign}, d_j, \mathcal{L})$ where:
     - c'_j is the assigned cluster
     - $\mathcal{T}_{assign}$ is the assignment task instruction
2. **Final Clustering:**
   - Documents are grouped based on assigned labels
   - Final clustering result: $\mathcal{C}' = \{c'_j\}_{j=1}^{K'}$

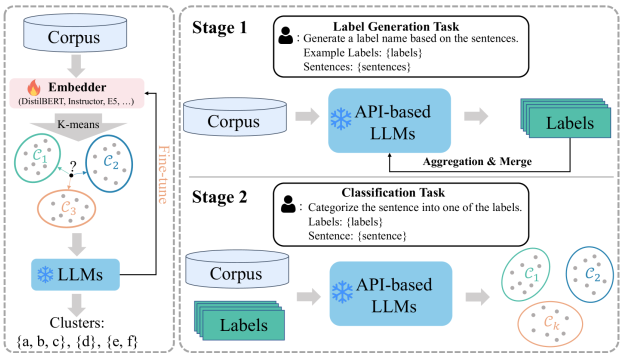

Figure 1: Other LLM Clustering methods (left side) and the solution from the paper [6] (right side)

2.1.1 Semantic Clustering with Sentence-BERT

Sentence-BERT (SBERT) [9] has significantly enhanced semantic clustering by creating sentence-level embeddings optimized for downstream clustering tasks. Unlike traditional BERT models, which are designed for token-level tasks, SBERT employs a siamese and triplet network structure to produce semantically meaningful embeddings.

In the study by Ortakci [8], researchers explored how different pooling strategies impact the clustering performance of SBERT embeddings. Pooling techniques like mean, max, and attention-based pooling were examined to optimize the representation of sentence-level information. The study demonstrated that leveraging pooling techniques in conjunction with SBERT embeddings significantly improves clustering methods like k-means and agglomerative clustering, leading to better semantic coherence within clusters.

2.1.2 Deep Embedded Clustering (DEC)

Deep Embedded Clustering (DEC) integrates feature learning with clustering by jointly optimizing latent space representations and cluster assignments. DEC employs autoencoders to reduce the dimensionality of textual data and refine clustering objectives. This approach enhances clustering accuracy and efficiency by dynamically adapting the latent space to the clustering task.

Xie et al. [14] proposed DEC to analyze high-dimensional text corpora, such as social media datasets, where the inherent noise and sparsity of data present challenges. By reconstructing the input data while simultaneously refining cluster centers, DEC effectively captures the underlying semantic structure, outperforming traditional clustering techniques in terms of both interpretability and precision.

2.1.3 Clustering with Large Language Models (LLMs)

The emergence of large language models (LLMs), such as GPT-3 and GPT-4, has opened new possibilities for clustering textual data. These models generate embeddings that encapsulate deep contextual and semantic information, which can be leveraged for clustering. Furthermore, LLMs facilitate an iterative feedback process to refine and validate cluster assignments.

Brown et al. [2] demonstrated the use of GPT-based embeddings for clustering scientific literature and business reports, achieving state-of-the-art performance in grouping semantically similar texts. Additionally, LLMs have been employed for cluster interpretation, where they assist in generating concise summaries or representative labels for each cluster, making the results more actionable.

2.1.4 Domain-Specific Clustering: Keyphrase Extraction

Domain-specific clustering approaches, such as keyphrase-based clustering, enhance the granularity of textual analysis by grouping texts based on semantically meaningful phrases. Keyphrase extraction techniques identify important terms that serve as features for clustering, effectively reducing noise in the data.

Bougouin et al. [1] proposed a hybrid approach combining TF-IDF and SBERT embeddings to organize large-scale news datasets by thematic content. Their approach outperformed generic clustering methods, particularly in domains where domain-specific terminology is prevalent.

2.1.5 Hybrid Approaches Combining Graph Neural Networks and NLP

Recent advances also include hybrid frameworks that combine graph neural networks (GNNs) with textual embeddings for clustering. These methods construct graphs where nodes represent textual data points and edges represent semantic similarities derived from embeddings. Clustering is then performed on the graph structure, leveraging community detection algorithms.

Kipf and Welling [7] introduced a graph-based clustering approach to analyze academic publications, where citation networks and textual content are jointly modeled. This method achieves higher precision in identifying research clusters compared to standalone embedding-based methods.

2.1.6 Comparative Evaluation of Clustering Techniques

Empirical studies have systematically compared the performance of these approaches across various datasets, such as scientific abstracts, customer reviews, and social media posts. Metrics like silhouette score, Davies-Bouldin index, and clustering purity indicate that embedding-based methods, particularly those utilizing SBERT, consistently outperform traditional feature-based clustering techniques. However, hybrid methods and LLMs exhibit superior performance in domains with complex semantic structures or large-scale datasets.

These metrics highlight the challenges of assessing clustering quality in the absence of definitive labels. Hybrid evaluation frameworks that combine quantitative scores with human validation can provide deeper insights into clustering effectiveness.

3 Datasets

For this study, we utilize several publicly available datasets that provide rich sources of textual data. The following datasets will be used for clustering and evaluation:

1. **Amazon Reviews:** The Amazon Reviews dataset is a comprehensive collection of product reviews across various categories, including electronics, books, and clothing. It includes text data, ratings, and metadata such

as product category and review date. This dataset is particularly useful for sentiment analysis and product categorization tasks. We will use reviews along with their ratings to explore clustering based on both sentiment and content. The dataset is available at: `https://nijianmo.github.io/amazon/index.html`.

2. **The 20 newsgroups text dataset:** The newsgroups dataset contains around 18,000 texts combined into 20 different clusters. It is a relatively small dataset; split into train and test dataset, which makes it a perfect sample for preliminary models testing. The dataset can be downloaded using scikit-learn API [12]: `https://scikit-learn.org/stable/datasets/real_world.html#the-20-newsgroups-text-dataset`

3. **BBC News dataset:** The BBC News dataset contains 2,225 documents divided into 5 categories: business, entertainment, politics, sport, and tech. Each document corresponds to a news article published by the BBC, with the labels representing the main topic of the article. This dataset is particularly useful for text classification tasks, such as topic modeling and supervised learning. Its relatively small size makes it an excellent choice for experimenting with models on clean and well-labeled text data. The dataset can be downloaded at: `http://mlg.ucd.ie/datasets/bbc.html`.

4 Solution Plan

4.1 Clustering Textual Data

For the clustering process, we propose two approaches.

1. Sentence-BERT (SBERT) Approach: The first approach will leverage Sentence-BERT (SBERT), a transformer-based model that generates semantically rich, dense embeddings for sentences or entire paragraphs. These embeddings capture deeper semantic relationships between sentences than traditional word embeddings. After obtaining the embeddings, we will experiment with various clustering algorithms, such as k-means, HDBSCAN and others. These algorithms will allow us to explore different clustering structures, with HDBSCAN in particular being useful for identifying clusters of varying densities, which is often ideal for textual data due to its inherent noise and varying distribution of content.

2. The second approach will combine Word2Vec embeddings with dimensionality reduction techniques such as UMAP (Uniform Manifold Approximation and Projection), autoencoders, etc, and clustering techniques such as k-means, HDBSCAN or etc. Word2Vec, a well-established word embedding technique, captures semantic relationships between words. By reducing the dimensionality of the word vectors with UMAP, we aim to visualize and cluster words or phrases in a lower-dimensional space. HDBSCAN will be employed to detect clusters of varying densities, which is particularly useful for discovering non-linear structures in textual data.

4.2 Evaluating Clustering Performance

4.2.1 Standard Quantitative Metrics

We will start by using well-established metrics from the state-of-the-art to evaluate clustering performance. These include:

- **F1-Score (F1S)** [4]: A well-known metric to balance between precision and recall. In the context of text clustering, it helps to ensure that clusters are both accurate and comprehensive.

- **Normalized Mutual Information (NMI)**: A metric that measures the agreement between the predicted clustering and the true labels, adjusting for chance. NMI is commonly used to evaluate clustering in settings where the true clusters are known and provides a normalized score between 0 and 1, with 1 indicating perfect agreement.

- **Adjusted Rand Index (ARI)** [13]: Another widely-used metric that compares the clustering results with the ground truth, while correcting for the possibility of random clustering. The ARI ranges from -1 (no agreement) to 1 (perfect agreement).

- **Silhouette Score** [11]: This score assesses the quality of clustering by measuring how similar an object is to its own cluster compared to other clusters. A high silhouette score indicates well-separated clusters.

- **Davies-Bouldin Index** [5]: This is another clustering evaluation metric that measures the average similarity ratio of each cluster with the cluster that is most similar to it. A lower score indicates better clustering quality.
- **Homogeneity score (HS)** [10]: A metric which allows to evaluate clustering quality by measuring how uniform a cluster is concerning a single class or label. It assesses whether all texts within a cluster share the same ground truth label. A score of 1 indicates perfect homogeneity, while a score of 0 suggests complete randomness.
- **Calinski-Harabasz Index (CHI)** [3]: A metric evaluating cluster coherence and separation of a datasets without the ground truth label. A higher value indicates better-defined clusters, with greater separation and tighter cohesion.

These metrics will help quantitatively assess the clustering quality and alignment with the ground truth, ensuring that our models are performing accurately.

Final Delivery

Introduction

After trying different techniques we decided to pivot from doing pure clustering into text classification. Table 2 shows the high-level comparison between classification and clustering particularly for textual data.

In order to cluster/classify 3 datasets we have, 2 approaches need to be considered:

1. For unlabeled data such as **Amazon Reviews**, we need to first generate labels for it and then assign. We implemented the paper "Clustering as Classification with LLMs" discussed in the next section, which helps us do this.
2. For the labeled datasets we used 2 approaches -
   - Implementing Stage 2 approach from paper "Clustering as Classification with LLMs"
   - Using regular machine learning such as classification of BERT embeddings.

5 Clustering Amazon Reviews

In order to cluster Amazon Reviews dataset the approach from the paper "Clustering as classification with LLMs" was used.

The pipeline was ran on a subset of the dataset, consisting of 10000 reviews. This was due to hardware limitation. The pipeline took about 6 hours to finish and after that each comment had a topic assigned to it.

5.1 Features of our implementation

In addition to getting topic assigned, we modified this pipeline to also be able to provide us with the sentiment of the comment. This could be helpful in multiple cases across different datasters. In the current dataset we also had information on the Rating the person left, which was used as "true labels". The rating was 1-5, so if the rating was 1-2, then the sentiment is negative. 3 - Neautral and 4,5 - positive.

The second modification which we did to the original paper was to use local (in-house) LLM. We used open source LLM - Gemini 9B, while the original paper used an API to GPT3.5 model. The API is not free, so our solution is truly open source

	Pros	Cons
Classification	• Proper classes make more sense in business scenarios. • Easily understood accuracy when training the model.	• Requires data labeling, which can be time-consuming and expensive.
Clustering	• No data labeling is needed, so any textual data can be used.	• No way of defining clusters automatically. • Reduced value for specific business scenarios.

Table 1: Comparison of Classification and Clustering

and can be run by anyone without any contributions.

As the result, taking the Amazon dataset with comments and ratings we were able to assess how well the model generates and assigns topics as well as gets the sentiment.

5.2 Results

In order to understand how to assess the assigned topic, we decided to label the data ourselves. We used statistics to label only a sample. Using the sample size formula for population (where size of population is 10000, since we need to get accuracy on 10000 predicted topics), we calculated that we need 370 samples.

$$n = \frac{Z^2 \cdot p \cdot (1-p)}{E^2}$$

- n: Required sample size
- Z: Z-score corresponding to the desired confidence level (e.g., 1.96 for 95% confidence)
- p: Estimated proportion of the population (assumed to be 0.5 if unknown, as it maximizes the required sample size)
- E: Margin of error (e.g., 0.05 for $\pm 5\%$)

Assumptions:

- **Confidence Level:** 95% ($Z = 1.96$)
- **Estimated Proportion:** $p = 0.5$
- **Margin of Error:** $E = 0.05$
- **Population Size:** $N = 10,000$

Hence if we plug in the numbers, we get that we need at least 369.98 (or 370) of samples to have 95% confidence that the calculated accuracy of predicted vs manually labeled data is correct.

Once the accuracy was checked on a sample of 370 comments, the accuracy of **97% was achieved**.

To get the result on the sentiment, below are all the metrics collected:

Metric	Value
Weighted Precision	0.858
Weighted Recall	0.875
Weighted F1 Score	0.854
Weighted Accuracy	0.903

Table 2: Evaluation Metrics

This shows that the LLMs can successfully predict the sentiment of the textual data while still assigning it to the topic.

5.3 Important Note

It's important to note that even thought the accuracy may have been very high, the process of generating topics was automated. And as the result of this, the topics were **very vague**, meaning that one topic could be applied to multiple comments such as ”Usage Experience” and ”User Experience”.

Another problem is that many comments are very constructive and they give points to multiple topics on purpose to address all aspects of the product.

To fix this issue, we encourage to

1. Generate topics automatically
2. Provide the topics to people from business and consult on which are useful and make sense. Ones these topics will readjusted, changed and/or merged, these topics could be used further on in the Stage 2 of the pipeline.

6 Classifying Labeled Datasets

6.1 LLM Classification

For both BBC News and 20 news groups datasets the same LLM architecture was used. Using Gemini 9B model, we passed information on both the comment and the potential labels on the test set. After this, the pipeline assigned the predicted label to each comment. We did not use any sentiment analysis on these datasets since they did not have true labels assigned.

Important note - training set was NOT used for this pipeline. This means that we could run this pipeline without any assigned labels, only on test set. It's vital to have labels themselves, which differs this approach from the full implementation of paper "Clustering as Classification with LLMs".

It's important to note that due to hardware limitations, the 20 news group dataset was ran only on 80% of the dataset. Nevertheless this was enough to get a good accuracy metrics in the results.

6.2 BERT Embeddings + SVM

For both BBC News and 20 news groups datasets the same architechture was used. In order to classify, we need to first embed the text data using RoBERTa embeddings. After multiple trials we got the best results with RoBERTa embeddings and no dimensionality reduction. Once the dataset was embedded, we trained SVM and tested it on the test set.

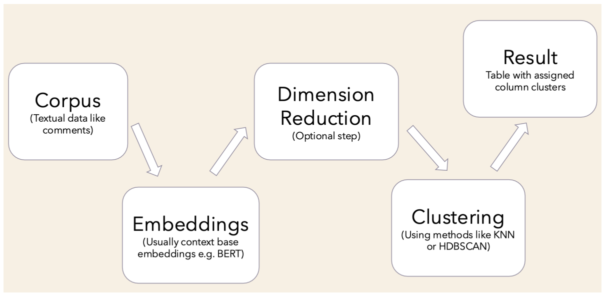
Corpus
(Textual data like comments)
Embeddings
(Usually context base embeddings e.g. BERT)
Dimension Reduction
(Optional step)
Clustering
(Using methods like KNN or HDBSCAN)
Result
Table with assigned column clusters

Figure 2: Pipeline to classify textual data

6.2.1 Results

The LLM approach runs on the test data directly while the SVM model is being trained. Nevertheless the approach with SVM is considerably faster. For 20 news group dataset, LLM approach was 6 hours, while the SVM pipeline took only 12 minutes. Most of this time was taken in order to generate RoBERTa embeddings.

Below are the tables for both datasets:

Accuracy Metrics	LLM	SVM
Weighted Accuracy	60.25%	64.17%
Weighted F1 Score	64.69%	64.06%
Weighted Precision	77.62%	64.43%
Weighted Recall	60.25%	64.17%

Table 3: Metric Results for 20News Group Dataset

Accuracy Metrics	LLM	SVM
Weighted Accuracy	94.97%	98.64%
Weighted F1 Score	94.93%	98.64%
Weighted Precision	95.16%	98.65%
Weighted Recall	94.97%	98.64%

Table 4: Metric Results for BBC News Dataset

As one could easily observe, the traditional SVM is much faster to run while still being able to outperform LLMs. This means that if there is a chance to use traditional ML (In case of classification of labeled data), it is always best to go with it.

The advantage LLM gives us is that we can label with very good accuracy without having any labels assigned (without training data).

7 Summary

LLMs provide great advantages in many aspects of NLP. In text clustering we can see that LLMs are capable of generating and assigning topics, and solving classification problems.

While both approaches work well, we recommend to beware of its limitations.

1. For unlaabeled data such as comments, the "Clustering as classification with LLMs" gives us a great overview. But it vital to consult with business on the stage of generating topics. For our example on amazon datasets some topics did not bring sensible results to business (e.g. topic - "Learning outcome" suggests that people who wrote a comment

on how they learned something is not exactly aplicable ot business).

2. For classifying the dataset having the labels it is possible to obtain additional information, such as sentiment and other interesting key information from the comment. At no cost it is possible to obtain additional information

3. If the additional information is not needed, as for future work we consider a good option to use SVM. One could first label the data using LLMs and afterwards train SVM on top of it to be able to predict the future classes without long runtime.

7.1 Reproducing Results

Our tests were run on Macbook Pro with M3 PRO chip and 18 GB of RAM. RAM is essential when running the LLM pipelines.
Hardware Requirements:

- CPU: High-performance CPU (like Intel i5)
- GPU: NVIDIA or AMD GPU with good graphics processing capabilities. Recommened GPU is RTX 3060.
- RAM: At least 16 GB
- Storage: Approximately 20 GB of storage for datasets and Gemini 9B model.

Software Requirements:

- Python3 installed.
- pip install requirements.txt file to get all the packages.
- Internet access.
- Donwload the datasets in the folder /datasets OR change the variable dataset_path in each notebook.

Notebooks:

- llm_amazon - pipeline to run notebook with implementation of the paper architecture for amazon clustering with additional sentiment analysis.
- llm_bbc - pipeline to run the LLM classification on BBC News dataset.
- llm_news - pipeline to run the LLM classification on 20 news group dataset.
- svm_bbc - pipeline to run the SVM classification on BBC News dataset.
- svm_news - pipeline to run the SVM classification on 20 news group dataset.

References

[1] Adrien Bougouin, Florian Boudin, and Béatrice Daille. Keyphrase extraction for clustering purposes: Combining textual and semantic features. *Journal of Information Retrieval*, 2021.

[2] Tom B. Brown, Benjamin Mann, and Nick et al. Ryder. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020.

[3] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[4] Nancy Chinchor and Beth M Sundheim. Muc-5 evaluation metrics. 1993.

[5] David L. Davis and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[6] Chen Huang and Guoxiu He. Text clustering as classification with llms. *StatNLP Research Group*, 2023.

[7] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations (ICLR)*, 2017.

[8] Yasin Ortakci. Revolutionary text clustering: Investigating transfer learning capacity of sbert models through pooling techniques. *Journal of Advanced Text Clustering*, 2023.

[9] N Reimers. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[10] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.

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

[12] Scikit-learn. The real world datasets.

[13] Douglas Steinley. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3):386, 2004.

[14] Jianlong Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. *Proceedings of the International Conference on Machine Learning*, 2016.