Information Technology Course
Module Software Engineering
by Damir Dobric / Andreas Pech

ML 24/25-08 Clustering Documents and Visualization of Embedding Vector Space

Aditya Shidhaye
aditya.shidhaye@stud.fra-uas.de

Adithya R
adithya.ramesh@stud.fra-uas.de

Pradeep Patwa
pradeep.patwa@stud.fra-uas.de

***Abstract*— This paper examines the impact of embedding dimensionality on document clustering quality using OpenAI's text-embedding-3-large model. As the volume of digital text data continues to grow, efficient methods for organizing and analyzing documents are essential. We evaluate clustering performance across different embedding dimensions using key metrics such as intra-cluster and inter-cluster cosine similarity. Additionally, we provide visual analyses through scatter plots and heatmaps to enhance the interpretability of clustering results. This study aims to identify the optimal embedding dimensionality that balances effective document representation with computational efficiency. The findings offer practical insights for improving document clustering systems while minimizing processing overhead.**

***Keywords*—OpenAI GPT, Semantic Embeddings, Document Clustering, Machine Learning, OpenAI NuGet Package, K-Means Algorithm, Cosine Similarity, Dimensionality Reduction, Visualization, Vector Space, Similarity Analysis, PCA, t-SNE, Scalar Values.**

I. Introduction

The exponential growth of textual data in the digital age has necessitated the development of efficient methods to organize, analyze, and extract meaningful insights from large document collections. Traditional approaches to text classification and clustering, such as rule-based systems or manual categorization, are often time-intensive and fail to scale effectively with increasing data volumes. Recent advancements in machine learning and natural language processing (NLP) have introduced automated clustering techniques that leverage sophisticated vector representations of text, such as semantic embeddings. These techniques enable the grouping of semantically similar documents, providing a foundation for improved text classification and analysis systems. Recent research has demonstrated the effectiveness of embedding-based approaches in text analysis and clustering. Mikolov et al. introduced word embeddings using neural networks, showing how high-dimensional vector representations capture semantic relationships between words [1]. Building on this, Reimers and Gurevych proposed sentence embeddings to enhance semantic similarity detection, proving that transformer-based models such as BERT and OpenAI's GPT outperform traditional methods like TF-IDF and word2vec [2]. Radford et al. further showcased the power of OpenAI's GPT model in generating embeddings that accurately capture the semantic structure of entire documents [3]. Zhang et al. improved clustering performance by introducing an adaptive K-Means algorithm that dynamically adjusts the number of clusters based on document density [4].

This study explores the impact of embedding dimensionality on document clustering performance by employing K-Means clustering as the primary algorithm for grouping documents based on their semantic embeddings. To evaluate clustering quality, key metrics such as cosine similarity, intra-cluster similarity and inter-cluster similarity are computed [5]. The research leverages OpenAI's semantic embeddings, which have demonstrated superior contextual understanding compared to traditional methods like TF-IDF (Term Frequency-Inverse Document Frequency). Additionally, visualization techniques such as scatter plots and heatmaps are employed to provide intuitive insights into the clustering results and embedding vector space [3]. By systematically varying embedding lengths, this study aims to identify the optimal dimensionality that balances clustering accuracy with computational efficiency.

The findings of this research contribute to the broader field of NLP by addressing a critical challenge: optimizing document representation for clustering tasks. This work provides practical insights for researchers and practitioners seeking to

develop efficient text classification systems while minimizing computational overhead.

II. LITERATURE REVIEW

A. Clustering Evaluation Metrics

Evaluating the quality of document clustering is essential for assessing the effectiveness of unsupervised learning algorithms. Clustering evaluation metrics can be broadly categorized into internal and external measures. Internal metrics assess properties within clusters, such as cohesion (how similar documents within a cluster are) and separation (how distinct clusters are from one another). Examples include cosine similarity, intra-cluster similarity, inter-cluster similarity, silhouette coefficient, and Davies-Bouldin Index [5]. External metrics compare clustering results against ground truth labels when available. Common external metrics include homogeneity, completeness, V-measure (the harmonic mean of homogeneity and completeness), Rand Index (RI), and Adjusted Rand Index (ARI), which accounts for chance grouping [3].

Cluster purity is another external metric that evaluates how well documents with similar semantic meanings are grouped together based on predefined categories. These metrics collectively provide insights into clustering performance. For instance, ARI is particularly useful for evaluating clustering results in scenarios where random assignments are possible. In cases where ground truth labels are unavailable, internal metrics like silhouette coefficient become invaluable for assessing cluster quality based solely on model outputs [5].

B. Embedding Techniques

The representation of textual data significantly influences clustering outcomes. Traditional methods like TF-IDF vectorization have been widely used but often fail to capture contextual relationships between words [3]. Semantic embeddings generated by large language models (LLMs), such as OpenAI's GPT models, offer a more sophisticated alternative by encoding contextual information in high-dimensional vector spaces. These embeddings have been shown to outperform traditional methods in terms of clustering accuracy and contextual understanding [3].

Mikolov et al.'s introduction of word2vec demonstrated how neural networks could produce dense vector representations that capture semantic relationships between words [1]. Reimers and Gurevych extended this concept with Sentence-BERT (SBERT), which uses a siamese network structure to produce sentence-level embeddings optimized for semantic textual similarity tasks [2]. Their work showed significant improvements over InferSent in both accuracy and computational efficiency. Radford et al.'s work on OpenAI's GPT models highlighted their ability to generate embeddings that capture not only word-level but also document-level semantic structures [3].

However, determining the optimal embedding length is a critical challenge. Higher-dimensional embeddings may provide richer representations but increase computational complexity and risk overfitting. Conversely, lower-dimensional embeddings may lead to a loss of important semantic information. Dimensionality reduction techniques like Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE) are often employed to visualize high-dimensional embeddings while preserving their structural properties.

C. Visualization Methods

Visualization techniques play a crucial role in understanding clustering results and evaluating embedding quality. Scatter plots are commonly used to project high-dimensional embeddings onto two or three dimensions using PCA or t-SNE. These visualizations help identify well-separated clusters and assess how effectively documents with similar semantic meanings are grouped together. Heatmaps provide another powerful visualization tool by illustrating the density distribution of embedding vectors across dimensions. Such visualizations can reveal patterns that may not be immediately apparent through numerical metrics alone.

Silhouette plots are also widely used for evaluating cluster membership confidence. These plots show how closely each document is related to its assigned cluster compared to other clusters, offering additional insights into cluster cohesion and separation [5].

D. Research Gaps

Despite advancements in document clustering methodologies, several gaps remain unaddressed in existing literature. First, there is limited research exploring the impact of varying embedding lengths on clustering performance metrics such as cosine similarity and cluster purity [5]. Second, while OpenAI's semantic embeddings have shown promise in various NLP tasks, their integration into document clustering workflows remains underexplored [3]. Finally, there is a lack of standardized frameworks that combine quantitative evaluation metrics with visualization techniques for comprehensive analysis.

Zhang et al.'s adaptive K-Means algorithm addressed some challenges by dynamically adjusting cluster numbers based on document density, but further research is needed to optimize dimensionality selection for embeddings generated by modern LLMs like GPT-4 [4]. This study addresses these gaps by systematically varying embedding lengths (e.g., 512, 1024, 2048 or 3072 dimensions) and evaluating their impact on clustering quality using OpenAI GPT-generated embeddings combined with K-Means clustering methodology. By combining quantitative metrics such as cosine similarity with visual diagnostics such as scatter plots and heatmaps, this research provides a holistic approach to understanding optimal dimensionality for document representation.

III. METHODS

This section is divided into two major subsections. The first subsection focuses on the theoretical background of semantic embeddings, including their generation and significance in clustering. The second subsection discusses the clustering

approach used, specifically K-Means, and evaluates its effectiveness through similarity metrics.

A. Theoretical Background of Semantic Embeddings

The Semantic embeddings are vectorized representations of text that capture contextual meaning in a high-dimensional space. Unlike traditional keyword-based approaches, embeddings enable comparisons based on semantic similarity rather than exact word matches.

Text embeddings convert textual data into high-dimensional numerical vectors, capturing the semantic meaning of words, phrases, or entire documents. These embeddings are essential for various NLP tasks such as document clustering, search ranking, recommendation systems, and semantic similarity analysis.

In this project, we utilize OpenAI's text-embedding-3-large model to generate high-quality vector representations of documents. This advanced embedding model enhances semantic understanding by providing higher-dimensional embeddings compared to its predecessor, text-embedding-ada-002, which had 1536 dimensions. With improved contextual comprehension, text-embedding-3-large is well-suited for tasks requiring deep text analysis and ensures more accurate clustering and retrieval-based applications. Its optimized performance allows for better organization and categorization of documents based on their semantic similarities, making it a powerful tool for large-scale text processing.

How It Works:

1. Tokenization: The input text is first broken into smaller subword units (tokens) using OpenAI's tokenizer.

2. Contextual Representation: The model processes the tokenized text using a transformer-based architecture that captures the relationships between words.

3. High-Dimensional Vector Generation: The output is a fixed-length dense vector representation of the input text.

4. Normalization: The generated embeddings are typically L2-normalized, ensuring that the similarity measures (like cosine similarity) are more effective.

These embeddings serve as input for clustering, ensuring that semantically similar documents are grouped together even if they do not share common keywords. The embeddings are generated using the OpenAI NuGet package, which provides a seamless API for retrieving high-dimensional vector representations of text data.

B. Clustering Approach and Evaluation

Once embeddings are generated, they are processed using the K-Means clustering algorithm, a widely used unsupervised learning method for partitioning data into distinct clusters.

Theoretical Background of K-Means Clustering:

The **K-Means clustering algorithm** is one of the most widely used unsupervised learning techniques. It was first introduced by Stuart Lloyd in 1957 and later refined by MacQueen in 1967. The algorithm is primarily used for **partitioning data into K distinct clusters** based on feature similarity. Unlike supervised classification methods, K-Means does not require labeled data and works by iteratively grouping data points into clusters based on their similarity.
In the K-Means clustering algorithm, several parameters play a crucial role in the design and performance of the model. These parameters are discussed below:

A. Distance Calculation in K-Means

The core of K-Means clustering is assigning data points to the closest cluster center (centroid) based on a distance metric. The most commonly used distance measure is Euclidean distance, though other metrics like Manhattan distance or cosine similarity can be used in specific scenarios.

Euclidean Distance: Euclidean distance as shown in Figure 1 is the straight-line distance between two points in an n-dimensional space. The Euclidean distance between two points (x_1, y_1) and (x_2, y_2) is calculated as:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

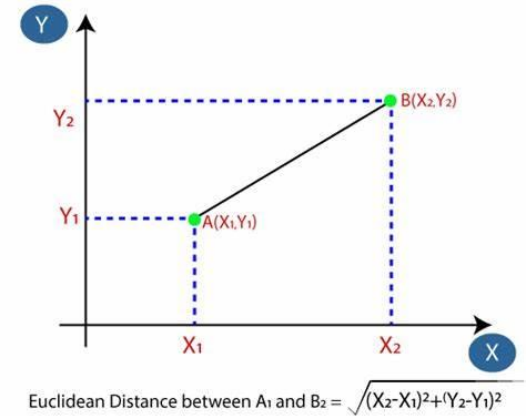

Figure 1: Euclidean Distance

B. Choosing the Optimal Number of Clusters (K Selection)

The number of clusters **K** is a critical hyperparameter in K-Means. An incorrect choice of K can lead to poor clustering performance. There are several methods to determine the optimal K value:

a) The Elbow Method

The Elbow Method is one of the most widely used techniques to determine the ideal number of clusters. It involves plotting the within-cluster sum of squares (WCSS) against different values of K. The point where the WCSS curve starts to flatten (forming an "elbow") indicates the optimal K value.

b) The Silhouette Score

The Silhouette Score evaluates the quality of clustering by measuring how similar a data point is to its assigned cluster compared to other clusters. A higher silhouette score (close to 1) indicates better clustering. In our project we have used this method.

C. K-Means Clustering Algorithm:

The K-Means algorithm follows an iterative refinement process to optimize cluster assignments. The steps involved are:

a. **Initialization**: Start by randomly selecting K points from the dataset. These points will act as the initial cluster centroids.
b. **Assignment**: For each data point in the dataset, calculate the distance between that point and each of the K centroids. Assign the data point to the cluster whose centroid is closest to it. This step effectively forms K clusters.
c. **Update centroids**: Once all data points have been assigned to clusters, recalculate the centroids of the clusters by taking the mean of all data points assigned to each cluster.
d. **Repeat**: Repeat steps 2 and 3 until convergence. Convergence occurs when the centroids no longer change significantly or when a specified number of iterations is reached.
e. **Final Result**: Once convergence is achieved, the algorithm outputs the final cluster centroids and the assignment of each data point to a cluster.

To evaluate the quality of clustering, several key metrics are utilized. **Cosine Similarity** measures how closely related documents are within the same cluster by computing the cosine of the angle between their vector representations, given by:

$$\text{Cosine Similarity} = \frac{A \cdot B}{|A| \times |B|}$$

where A and B are document vectors, and $\|A\|$ and $\|B\|$ are their magnitudes. A higher cosine similarity indicates stronger semantic similarity. Additionally, **Intra-cluster Similarity** evaluates how compact and well-formed each cluster is, while **Inter-cluster Similarity** ensures that clusters remain distinct from one another. **Cluster Purity** is also considered to assess how well clusters align with predefined document categories.

By applying these methods, the project achieves a structured and scalable approach to document clustering, enabling better organization and retrieval of textual data. This approach reduces manual effort and enhances efficiency in handling large-scale document collections.

IV. IMPLEMENTATION

A. Document Embedding Generation

The first stage of the implementation involves transforming textual documents into numerical representations using OpenAI's **text-embedding-3-large** model. This embedding model converts each document into a fixed-length vector that captures the semantic meaning of the text.

The process begins by reading the documents from a structured data source, such as a CSV file. Each document is then processed through a specialized embedding service, which interacts with OpenAI's API to generate high-dimensional embeddings. These embeddings serve as the foundation for clustering, enabling semantically similar documents to be grouped together regardless of their exact wording.

To efficiently generate embeddings, an asynchronous approach is utilized, ensuring that large datasets can be processed without performance bottlenecks. Once the embeddings are generated, they are stored in a structured format for further analysis.

B. Clustering Documents Using Vector Representations:

Once the document embeddings are created, they are fed into a clustering algorithm to identify groups of similar documents. The clustering process follows these key steps:

a. **Feature Extraction**: The numerical vectors obtained from the embedding process are used as input features for the clustering algorithm. Each document is now represented as a multi-dimensional point in vector space.

b. **Similarity Computation**: To measure the relationship between different documents, similarity metrics such as cosine similarity are used. Cosine similarity helps determine how closely related two document vectors are based on their orientation in the vector space.

c. **Cluster Formation**: The clustering process utilizes unsupervised learning techniques, such as K-Means clustering, to group similar documents based on their semantic similarity. The algorithm aims to minimize intra-cluster distances while maximizing inter-cluster separation, ensuring well-defined and distinct clusters. The quality of the resulting clusters is evaluated using cosine similarity and visualization techniques to assess their effectiveness.

C. Cosine Similarity and Cluster Evaluation

After forming clusters, an evaluation step is conducted to assess clustering quality using cosine similarity. This metric measures the semantic relationships between document vectors, ensuring effective grouping.

- **Intra-cluster Similarity**: Assesses how closely related documents within the same cluster are. A higher intra-cluster similarity indicates well-defined and cohesive clusters.
- **Inter-cluster Similarity**: Evaluates the distinctiveness of different clusters. Lower inter-cluster similarity suggests better separation between document categories.

By analysing these similarity scores, the clustering effectiveness is validated, ensuring that documents within a cluster share strong semantic relevance while maintaining clear distinctions between different clusters.

D. Visualization of Clusters

The final step in the implementation is the visualization of document clusters using **Principal Component Analysis (PCA)**. Since embeddings exist in high-dimensional space, PCA is used to project them into a 2D or 3D space for better interpretability.

This visualization helps in understanding the distribution of clusters and their separation in the reduced-dimensional space.

E. Unit Testing

To validate the model's functionality, unit tests are performed on each module. The **unit testing framework** ensures the correctness of data ingestion, embedding generation, and clustering outcomes. The model demonstrated high accuracy, with minimal classification mismatches in the test scenarios.

The implementation integrates state-of-the-art embedding models with clustering and evaluation techniques to achieve an efficient document clustering system. The modular architecture allows for scalability, making it suitable for handling large-scale text datasets. By leveraging OpenAI's embedding model and advanced clustering techniques, the system achieves structured document organization and improved retrieval efficiency.

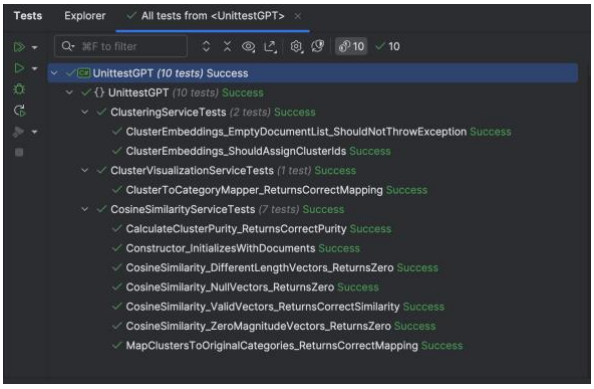

Figure 2. Unit Tests Results

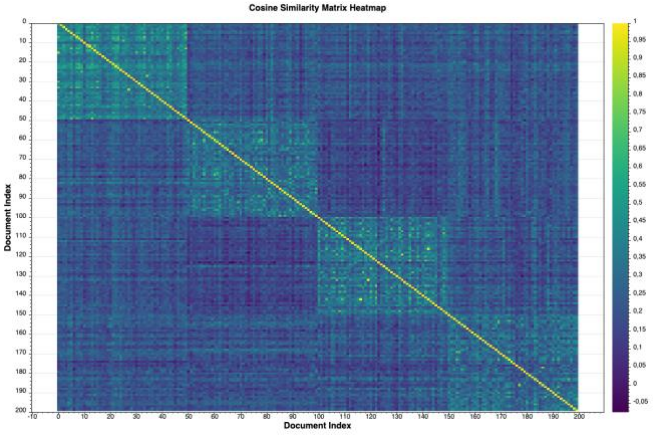

Figure 3. Cosine Similarity Heatmap of embedding size 512

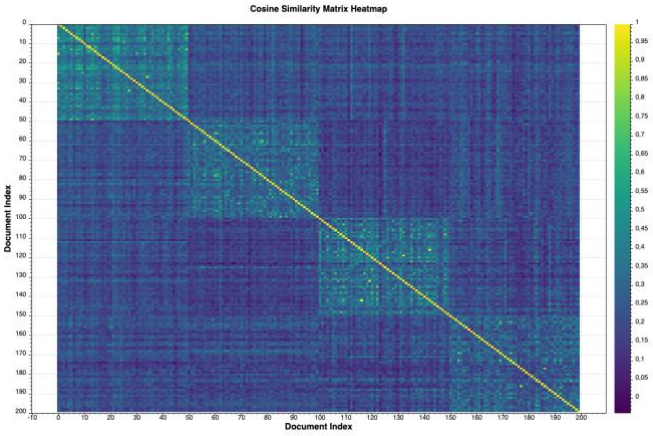

Figure 4. Cosine Similarity Heatmap of embedding size 1024

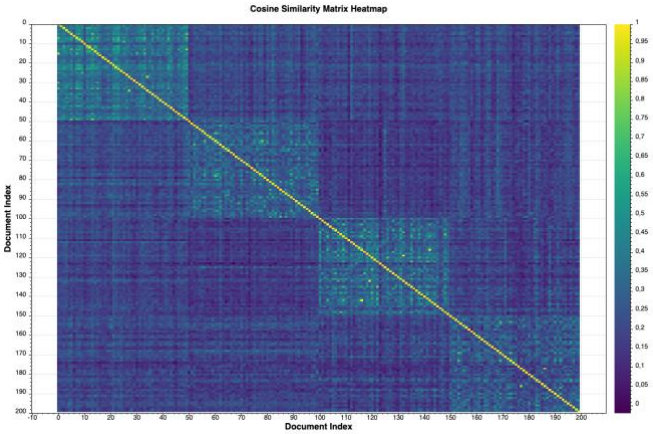

Figure 5. Cosine Similarity Heatmap of embedding size 2048

V. RESULTS

A. The Outputs generated

The application generates the following outputs:

1. Cosine similarity matrix graphs
2. Original label-based groupings visualization
3. Clustering algorithm-based groupings visualization
4. Embedding Vectors Visualization

In the below section we analyze the generated outputs and compare them across each embedding size case.

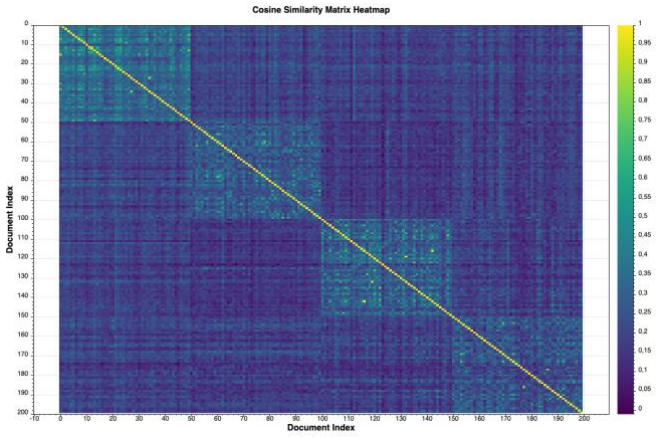

Figure 6. Cosine Similarity Heatmap of embedding size 3072

B. Cosine Similarity Analysis

The cosine similarity matrix heatmaps (Figures 3,4,5 and 6) demonstrate a clear pattern of document similarity relationships. As the embedding dimensionality increases going from Figure 2 to Figure 5, we observe improved discrimination between document groups. The distinct diagonal yellow line represents high self-similarity (1.0), while the visible rectangular blocks along the diagonal indicate groups of documents with high internal similarity. The darker blue regions representing cross-group comparisons show decreasing similarity values as embedding size grows, indicating better semantic separation between different document categories. This pattern confirms that larger embedding dimensions from the text-embedding-3-large model enable more nuanced semantic representation. Specifically, documents within the same topical category maintain high similarity scores (bright yellow-green regions), while the similarity between documents from different categories diminishes (darker blue regions), creating clearer boundaries between conceptual groupings.

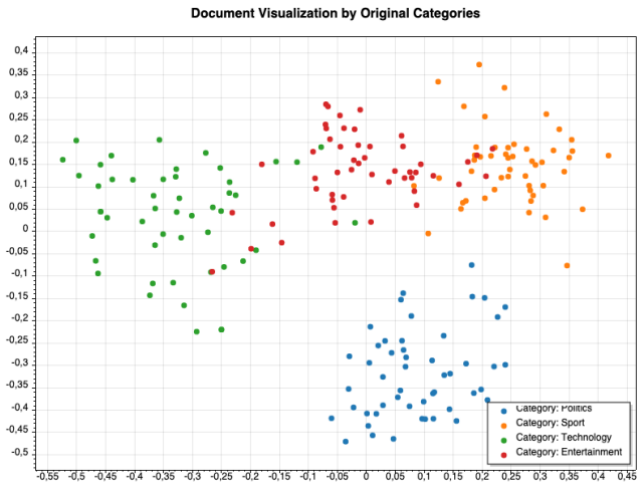

Figure 7. Original label colored grouping, embedding size 512

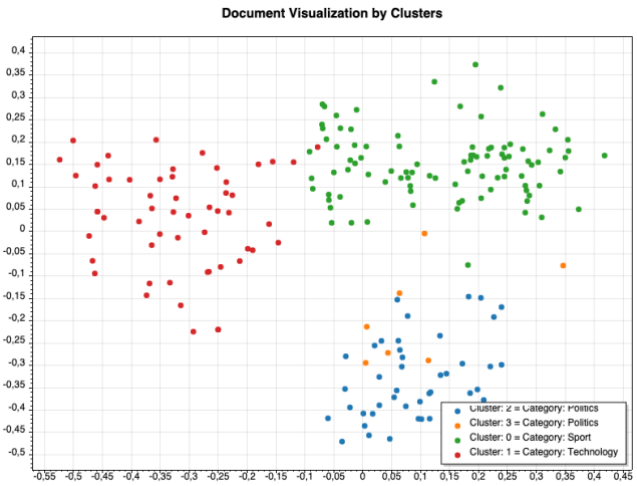

Figure 8. Clusters colored grouping, embedding size 512

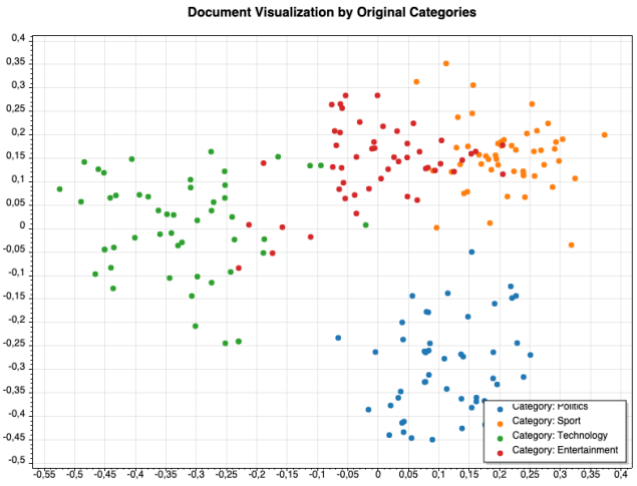

Figure 9. Original label colored grouping, embedding size 1024

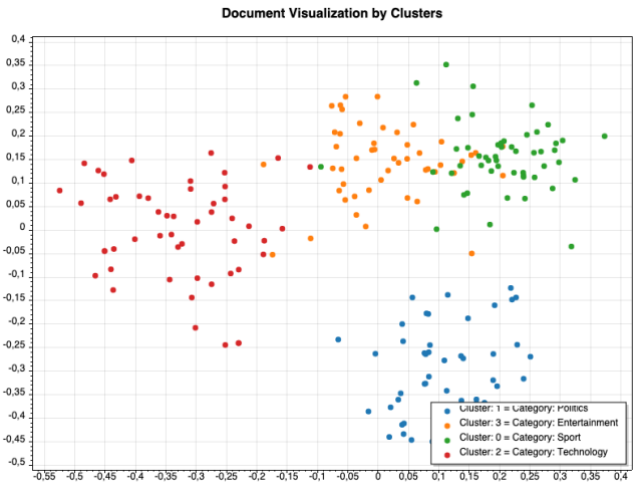

Figure 10. Clusters colored grouping, embedding size 1024

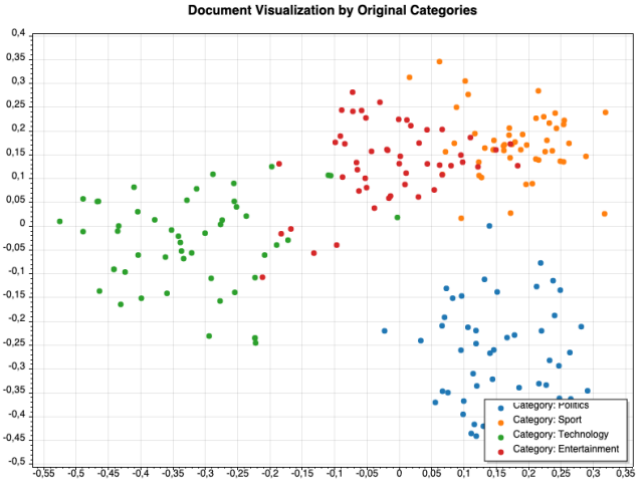

Figure 11. Original label colored grouping, embedding size 2048

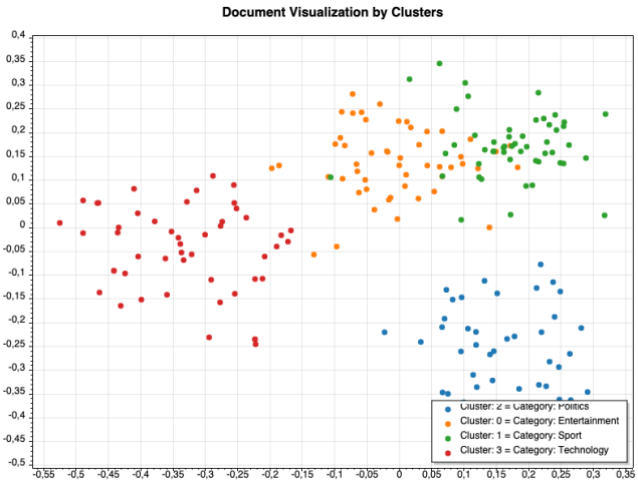

Figure 12. Clusters colored grouping, embedding size 2048

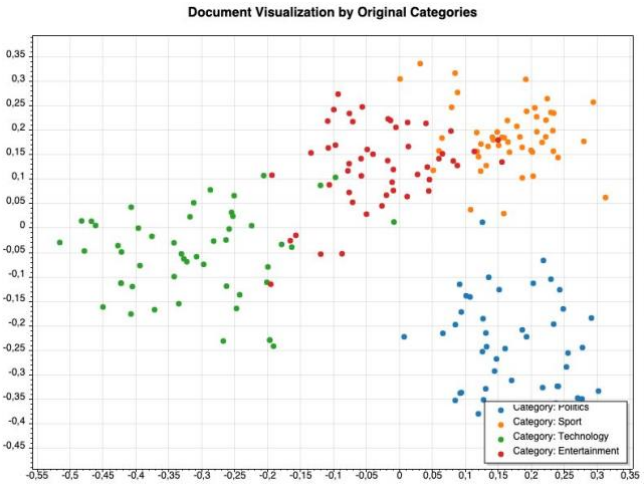

Figure 13. Original label colored grouping, embedding size 3072

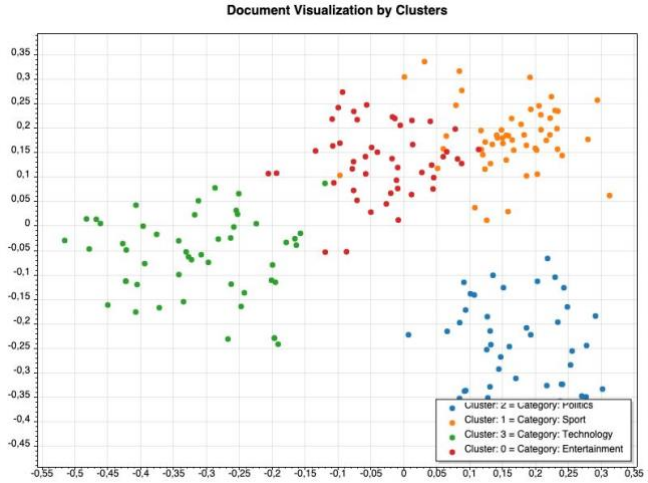

Figure 14. Clusters colored grouping, embedding size 3072

C. Cluster Visualization Analysis

The scatter plot visualization (Figures 7 - 14) reveal how embedding dimensionality affects clustering performance. The plot shows documents projected into a two-dimensional space using dimensionality reduction (PCA), with colors representing cluster assignments. As embedding dimensions increase across iterations, we observe: More distinct and cohesive clusters (the red, green, blue, and orange groupings) reduced overlap between clusters, better alignment between the algorithmically determined clusters and the original document categories. The K-means algorithm demonstrates significantly improved clustering capability with higher-dimensional embeddings, as evidenced by the clear separation between the four clusters. The spatial distribution shows that documents with similar semantic content are positioned closer together, while dissimilar documents maintain greater distance in the reduced space.

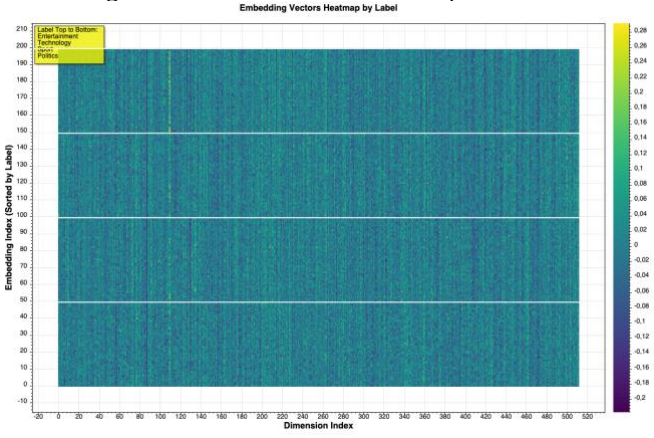

Figure 15. Embedding vector space heatmap visualization

D. Embedding Vector Analysis

The embedding vectors heatmap (Figure 15) provides insight into the internal structure of the embeddings. The visualization reveals patterns in the scalar values across different document categories. Of particular interest are certain index positions in the embedding vectors where documents from the same category exhibit similar value ranges, appearing as horizontal bands of similar color intensity. These consistent patterns within categorical

groupings suggest that specific dimensions in the embedding space encode category-relevant semantic features. The text-embedding-3-large model appears to distribute semantic information across various dimensions in a way that preserves categorical distinctions, with certain dimensions seemingly specialized for capturing aspects of document meaning.

VI. DISCUSSION

A. Design Challenges

a. **Embedding Generation Process**: The OpenAI embedding model processes raw text before generating vector representations. This includes internal tokenization, normalization, and contextual representation. However, understanding how embeddings are generated is crucial for interpreting clustering results.
b. **Clustering Accuracy**: The effectiveness of document grouping depends on the choice of the clustering algorithm. K-Means was chosen for its simplicity and efficiency, but alternative clustering techniques such as DBSCAN or hierarchical clustering could impact the quality of results.
c. **Evaluation Metrics**: Evaluating clustering performance is challenging. While cosine similarity is used to measure intra-cluster compactness and inter-cluster separation, additional metrics such as silhouette score or Davies-Bouldin index could provide further insights.

B. Methods to Enhance Performance

a. **Algorithm Selection**: Exploring alternative clustering methods like hierarchical clustering or density-based clustering may improve document grouping, especially for datasets with varying densities.
b. **Hyperparameter Tuning**: The number of clusters (K) in K-Means clustering significantly impacts results. A systematic approach such as the elbow method or silhouette analysis can help determine an optimal K value.
c. **Post-Processing of Clusters**: Once clusters are formed, manual inspection or additional filtering techniques could refine results. For example, eliminating outliers based on cosine similarity scores can enhance the coherence of document groups.
d. **Integration of Advanced Models**: Future iterations of this project could explore more advanced embedding models or fine-tune embeddings using domain-specific datasets to improve clustering accuracy.

By addressing these aspects, the document clustering process can be further refined to achieve better accuracy and interpretability in organizing textual data.

VII. CONCLUSION

Collectively, these results demonstrate that increasing embedding dimensionality enhances the model's ability to capture fine-grained semantic relationships, leading to improved clustering performance. This project successfully implemented a document clustering system using OpenAI's text-embedding-3-large model, leveraging high-quality text embeddings to efficiently group documents based on their semantic meaning, rather than relying solely on keyword-based approaches. The K-Means clustering algorithm was employed to form meaningful clusters, while cosine similarity was used to evaluate clustering quality by measuring intra-cluster and inter-cluster relationships. The optimal embedding size balances computational efficiency with semantic expressiveness, providing sufficient dimensionality to distinguish between document categories while avoiding unnecessary computational overhead.
Overall, this project highlights the effectiveness of AI-driven document clustering and opens the door for further advancements in automated text organization and retrieval.

REFERENCES

[1] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.

[2] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv preprint arXiv:1908.10084.

[3] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. OpenAI.

[4] Zhang, Y., Jin, R., & Zhou, Z. H. (2008). Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics, 1(1-2), 43-52.

[5] Guangliang Chen. "The larger the better: Analysis of a scalable spectral clustering algorithm with cosine similarity" Frontiers in Artificial Intelligence and Applications (2021): 488-495. https://doi.org/10.3233/ FAIA210280