Clustering Software Engineering Research with Text Embeddings

Turner Burchard
Computer Science Department
Montana State University
Bozeman, Montana
turnerburchard@gmail.com

***Abstract*—Most modern research fields now contain thousands of papers, often written on a diverse set of topics. These papers tend to be disorganized and spread across numerous journals and conferences, making them difficult to explore for researchers and professionals. To address this, we employ text embedding and clustering techniques to systematically organize software engineering literature into semantically coherent categories. By applying a streamlined dimensionality reduction step and evaluating multiple clustering algorithms, we found that K-means and Gaussian Mixture Models consistently outperformed alternatives. These methods produced well-separated, meaningful clusters, offering a data-driven framework for rapidly gaining insights into the evolving state of software engineering research.**

***Index Terms*—software engineering, text embedding, dimension reduction, clustering algorithms, research categorization, unsupervised learning, empirical analysis**

I. Introduction

The rapid expansion of research across modern fields has led to a proliferation of academic papers, often numbering in the thousands within a single domain. This explosion of information presents a significant challenge: researchers and professionals alike struggle to navigate the breadth of knowledge effectively. The problem is particularly acute in software engineering, where research spans a wide array of topics and is often scattered across diverse journals and conferences. The lack of organization hinders the accessibility and utility of these works, particularly for professionals seeking actionable insights.

To address this challenge, it is essential to develop systematic methods for reviewing and categorizing research. Recent advances in text embedding techniques offer a promising solution. By encoding textual information into numerical representations, text embeddings enable the application of machine learning methods, such as clustering, to group related works based on their content. These tools have reached a level of maturity that makes them suitable for application to complex, real-world datasets.

In this paper, we leverage text embeddings and clustering algorithms to classify software engineering research into distinct categories. The resulting categorization facilitates the exploration of the field and supports targeted inquiries into specific areas of interest. Furthermore, we introduce a basic tool that could provide rapid summarizations of research topics, offering a practical solution for researchers and professionals to understand the state of the field. This approach not only aids in organizing the existing literature, but also lays the groundwork for more structured and accessible research.

A. Background

Vector embeddings, also referred to as word embeddings in the context of natural language processing, are numerical representations of textual data designed to capture semantic meaning in a continuous vector space [1]. These representations address the limitations of earlier techniques, such as the bag-of-words model, which failed to account for semantic relationships between terms [2]. Embeddings encode words or documents as dense vectors in a high-dimensional space, where semantically similar entities are positioned closer together. This is achieved by training models on large corpora of text, leveraging co-occurrence statistics and contextual information [3].

Early methods such as Word2Vec, introduced by Mikolov et al., used shallow neural networks to predict context given a word (skip-gram) or a word given its context (CBOW) [4]. GloVe (Global Vectors for Word Representation) further advanced the field by combining global co-occurrence statistics with a local context window to produce embeddings that balance efficiency and accuracy [5]. These approaches demonstrated the power of embeddings in capturing complex linguistic structures, such as analogical reasoning (e.g., ”king - man + woman = queen”) [6].

Recent advancements, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), have introduced contextual embeddings that dynamically adjust based on surrounding words, improving understanding of polysemous terms [7]. These methods have broadened the applicability of embeddings to domains beyond text, such as image analysis and recommendation systems [8].

In this work, text embeddings are utilized to encode the abstracts and titles of research papers, providing the foundation for clustering algorithms to group similar works. This approach preserves semantic information and enables the discovery of meaningful relationships between research topics, making embeddings an effective tool for organizing large, unstructured datasets like software engineering research.

Clustering methods, as unsupervised machine learning techniques, identify natural groupings within data by optimizing specific criteria, such as minimizing intra-cluster variance while maximizing inter-cluster separation [**?**]. These methods are widely used in applications such as biology, marketing, and text analysis, where they reveal hidden structures within datasets [9]. Common clustering methods include K-means, DBSCAN, and hierarchical clustering, each with unique strengths and limitations based on the dataset's characteristics.

B. Motivation

Despite the advancements in text embedding and clustering techniques, their combined application to research categorization remains underexplored, possibly due to the complexity of interpreting high-dimensional embeddings. However, research has shown that embeddings capture meaningful semantic relationships [10], and clustering methods have successfully applied these properties in diverse areas such as social media analysis [11], categorization of large language model responses [12], and classification of software applications based on textual descriptions [13]. This study aims to bridge this gap by applying these techniques to the organization and analysis of Software Engineering research papers, leveraging the strengths of both embeddings and clustering to uncover meaningful patterns in the literature.

C. Research Hypotheses

Hypothesis: Clustering algorithms applied to embedded text representations of Software Engineering research papers will produce distinct and meaningful clusters, and at least one clustering algorithm will perform significantly better than the others in terms of producing well-separated clusters as measured by clustering validity metrics.

Exploratory Goal: The clusters produced by the algorithms are expected to align with meaningful semantic categories in the domain of Software Engineering research. While not directly testable, this alignment will be qualitatively assessed to provide insights into the practical utility of the clustering approach.

II. Approach

Our overall goal for this experiment requires several steps of collection, processing, and analysis. Specifically, we began by collecting a large sample of relevant research papers programmatically. This data was then embedded with a small vector embedding tool, in order to have numerical representations of the data. This data was then reduced in dimension for simpler and more effective clustering. Several clustering techniques were then applied to the data, along with the required hyperparameter tuning. Finally, we took several steps of analysis to understand which of the clustering techniques was best, and whether the process as a whole was effective.

A. Data Collection

To facilitate the clustering of software engineering research papers, the first step involved collecting a suitable dataset. Given the scale of the study, manual acquisition of research papers was impractical. Instead, automated APIs were employed to efficiently gather a large number of research papers along with their metadata. Crossref was identified as an ideal source due to its extensive database, comprising over 100 million records, including several million in computer science. From this dataset, we targeted papers relevant to software engineering.

The Crossref API enabled programmatic retrieval of metadata for relevant papers, streamlining the data collection process. To address computational constraints, a random sample of 500 papers per test was selected, with 10 tests conducted for each clustering method, resulting in a dataset of 5000 paper samples. However, as sampling was performed with replacement, the total unique number of papers analyzed is likely less than 5000.

To ensure relevance, the search was restricted to papers published within the last five years. The search criteria included "Software Engineering" with manual filtering and adjustments to exclude unrelated papers wherever possible.

B. Vector Embedding

To embed the textual data, we selected a single, efficient embedding model. This choice was guided by the computational constraints of the dataset and the high cost of using large embedding algorithms. A model with fewer dimensions was preferred to simplify clustering, as it is computationally more efficient to use low-dimensional embeddings directly rather than reducing high-dimensional embeddings post hoc. Additionally, the ability to run the embedding model locally was crucial to minimize costs, reduce network latency, and improve overall efficiency.

We opted to use FastEmbed, a 384-dimensional embedding model that is both fast and has been shown to be more accurate than certain larger, more complex models [14]. Its lower dimensionality not only enhances computational efficiency but also makes it particularly well-suited for clustering tasks, where simplicity and interpretability are critical. Additionally, its native Python library provides straightforward implementation, making it a practical and effective choice for this study.

C. Dimension Reduction

Principal Component Analysis (PCA) is a standard technique for unsupervised dimensionality reduction that identifies directions of maximal variance in high-dimensional data. Applying PCA to our 384-dimensional embeddings allowed us to emphasize the most informative components and discard redundancy. Although embeddings often contain valuable information across all dimensions, prior work [15] demonstrates that PCA can balance dimension reduction and clustering quality. Guided by these findings, we tested various target dimensionalities and variance thresholds before settling on a configuration that retained approximately 70% of the variance. This choice was determined to be an effective balance between complexity and the quality of downstream clustering. Figure 1

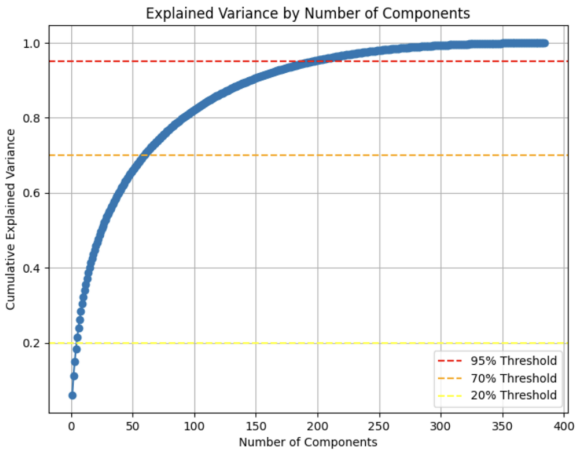

Fig. 1. Explained variance by the number of PCA components.

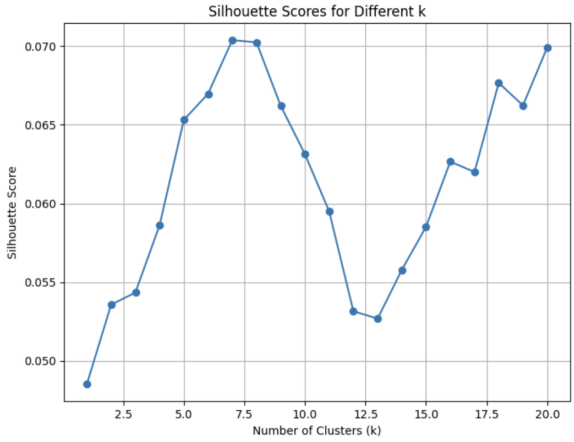

Fig. 2. Silhouette scores as a function of the number of clusters for K-means clustering.

illustrates the cumulative variance explained by increasing the number of principal components.

D. Clustering

After embedding the textual data and reducing its dimensionality, clustering is performed to analyze the structure of the data. Various modern clustering algorithms were evaluated to identify the one that optimally separates clusters. The methods considered include K-means, MiniBatch K-means, DBSCAN, Agglomerative Hierarchical Clustering, Self-Organizing Maps (SOM), Birch, and Gaussian Mixture Models.

Each algorithm requires tuning specific hyperparameters, with the number of clusters being a common parameter across many methods. A grid search approach was employed for hyperparameter tuning, initially using a broad range of values and refining these based on observed results. To validate the clustering performance, results were projected into two-dimensional space and visualized as scatter plots with distinct colors for each cluster. These visualizations helped assess the separability of clusters and guided adjustments, particularly to the number of clusters.

Despite efforts to fine-tune hyperparameters, achieving perfect optimization was challenging due to the unsupervised nature of the problem, the variety of algorithms, and the dataset's size. However, the tuning process followed practices commonly observed in the literature and produced results significantly better than random parameter selection.

Figure 2 illustrates the relationship between the number of clusters and silhouette scores for K-means clustering, highlighting the impact of cluster selection on performance.

E. Analytical Methods

Analyzing the clustered data requires a careful approach due to the unsupervised nature of the problem, which inhibits direct evaluation of clustering accuracy. To address this, silhouette scores are employed as a primary metric to assess cluster compactness and separation. This provides an objective basis for comparing the performance of different clustering techniques and identifying those that produce the most cohesive and well-separated groupings.

Beyond these metrics, statistical methods are utilized to compare the robustness of clustering results across techniques. One-way ANOVA and post hoc analyses are conducted to identify significant differences in performance metrics. Statistical significance testing ensures that observed differences are not attributable to random variation, thereby enhancing the reliability of the results.

To evaluate the practical utility of the clusters, interpretability and alignment with known structures in the software engineering domain are assessed. This includes a qualitative analysis of the clusters, examining whether they correspond to meaningful subfields, themes, or topics within the literature. Domain-specific validation further grounds the findings, ensuring that the identified clusters have relevance and coherence within the field.

III. Results

A. Clustering Performance

Table I summarizes the silhouette scores for the seven clustering algorithms tested at 50 dimensions across 10 trials. K-means and Gaussian Mixture models demonstrated the highest average silhouette scores, suggesting superior cluster separability. In contrast, Birch consistently showed the lowest scores, indicating weaker clustering performance.

B. Cluster Visualization

Figure 3 depicts an example of K-means clustering results projected into two dimensions using principal component analysis. Clusters were manually labeled to illustrate semantic separation. Visual inspection supports the numerical findings, confirming well-separated clusters for K-means.

C. Statistical Analysis

All clustering algorithms were evaluated across multiple trials using quantitative metrics. To assess whether the choice of clustering algorithm influenced silhouette scores, we first

TABLE I
CLUSTERING ALGORITHM SILHOUETTE SCORES

Algorithm	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9	Trial 10	Average
K-means	0.0731	0.0749	0.0775	0.0790	0.0776	0.0772	0.0761	0.0768	0.0748	0.0748	0.0762
Minibatch K-means	0.0555	0.0594	0.0600	0.0621	0.0606	0.0624	0.0640	0.0633	0.0641	0.0664	0.0618
DBSCAN	0.0501	0.0507	0.0544	0.0532	0.0547	0.0557	0.0529	0.0518	0.0561	0.0583	0.0538
Agglomerative Hierarchical	0.0535	0.0546	0.0543	0.0526	0.0543	0.0533	0.0501	0.0507	0.0525	0.0527	0.0529
SOM	0.0554	0.0563	0.0547	0.0537	0.0538	0.0545	0.0562	0.0560	0.0559	0.0548	0.0551
Birch	0.0384	0.0355	0.0351	0.0325	0.0330	0.0318	0.0322	0.0325	0.0333	0.0362	0.0340
Gaussian Mixture	0.0722	0.0741	0.0762	0.0752	0.0763	0.0767	0.0745	0.0739	0.0720	0.0736	0.0745

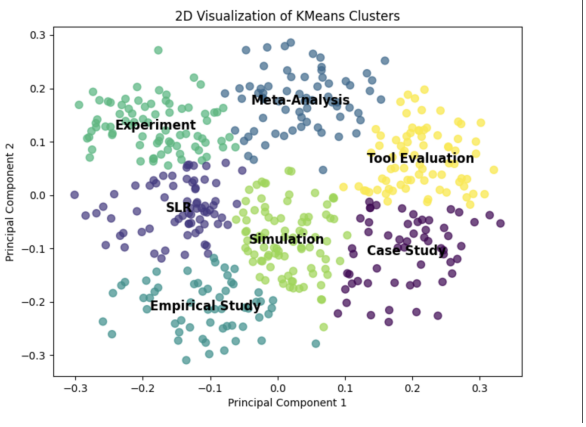

Fig. 3. Example of K-means clustering results projected to 2 dimensions with manual semantic labeling.

verified the normality of residuals via the Shapiro-Wilk test (p ¿ 0.05) and confirmed homogeneity of variance through Levene's test. We then conducted a one-way repeated measures Analysis of Variance (ANOVA), treating the algorithm as a within-subject factor and silhouette scores as repeated measurements. The ANOVA showed a significant main effect of algorithm on clustering performance (F(6, 54) = 479.30, p ¡ 0.0001).

To determine which algorithms differed significantly, we employed Tukey's Honestly Significant Difference (HSD) post hoc test. Results revealed that K-means and Gaussian Mixture Models both produced significantly higher silhouette scores than the other methods (p ¡ 0.05), with no significant difference between these two top performers.

IV. DISCUSSION

This study aimed to assess the performance of various clustering algorithms in identifying meaningful groups within a corpus of Software Engineering research papers. The results of the statistical analysis, including the ANOVA and post hoc tests, provide significant insights into algorithm performance and the structure of the data.

The repeated measures ANOVA revealed significant differences between clustering algorithms ($F(6, 54) = 479.3016, p < 0.0001$). Post hoc Tukey HSD analysis confirmed that K-means and Gaussian Mixture algorithms performed significantly better than other methods with 95% confidence. This result supports our hypothesis that certain algorithms are better suited for identifying separable clusters in high-dimensional textual data.

Despite their relatively low silhouette scores due to the curse of dimensionality, K-means and Gaussian Mixture excelled in discovering clusters with clear semantic meanings. These algorithms' superior performance is likely because they assume spherical clusters, which aligns with the structure of the research data. DBSCAN, by contrast, is designed for arbitrarily shaped clusters, such as rings or elongated structures, which are less likely to represent the topics in Software Engineering research. Gaussian Mixture's use of soft clustering likely enhances its ability to handle overlapping clusters, explaining its strong performance.

Manual inspection of the clusters revealed consistent topics present in the clusters. This inspection was done by manually checking the points within a cluster, and verifying whether the clusters were semantically coherent. We identified stable topic clusters such as case studies, heuristic methods, frameworks, systematic literature reviews, and novel tools. These results suggest that the vector space effectively encodes meaningful information, though some noise was present, with papers that span multiple topics (e.g., an SLR on case studies) occupying intermediate positions in the space. This overlap indicates that the high-dimensional vector space preserves nuanced relationships that could be further exploited with more advanced techniques.

While K-means and Gaussian Mixture demonstrated strong performance, the utility of other methods should not be overlooked. Agglomerative Hierarchical Clustering, while underperforming numerically, offers significant interpretive advantages due to its hierarchical structure. This technique enables the discovery of nested clusters, such as "Computer Science" containing "Software Engineering," and further sub-clusters like those identified here. This hierarchical representation provides a broader context for understanding the relationships among topics and is particularly useful when analyzing larger datasets.

Looking ahead, custom-built clustering methods designed for high-dimensional, noisy, and overlapping spheroids may improve performance further. Such methods could leverage domain-specific insights and advanced statistical techniques to enhance cluster separation and interpretability.

In summary, K-means and Gaussian Mixture algorithms successfully identified meaningful clusters, but further refinement of clustering approaches may be necessary to fully exploit the latent structure of the data and address its inherent noise and overlap.

V. Conclusion

The results of our repeated measures ANOVA and post hoc analysis provide support for our hypothesis. Clustering algorithms such as K-means and Gaussian Mixture significantly outperformed others, demonstrating their effectiveness in producing distinct and meaningful clusters aligned with the semantic content of research papers. However, the variability in performance across algorithms indicates that no single method fully captures the complex semantic structure inherent in Software Engineering research papers. Future exploration, such as parameter tuning or hybrid approaches that combine multiple clustering techniques, may improve clustering quality and consistently uncover meaningful patterns.

These findings highlight the importance of selecting the appropriate clustering algorithm for specific research contexts. K-means performed well due to its assumption of spherical clusters, which aligns with the nature of this dataset. Similarly, Gaussian Mixture excelled due to its use of soft clustering, which enhanced its ability to handle overlapping clusters. On the other hand, Agglomerative Hierarchical Clustering, while numerically underperforming, offers unique advantages in its ability to represent nested relationships, providing additional insights in larger-scale datasets.

A. Threats to Validity

Although the study was carefully designed, several potential threats to validity should be considered:

1) Testing Validity: Errors in data processing, automation, or sampling could impact the accuracy of the results. For instance, automating the extraction of textual data may lead to the over- or underrepresentation of certain information.

2) Internal Validity: Bias in Data Collection: The Crossref API, while extensive, is not comprehensive. Research that is not uploaded to Crossref is excluded, potentially omitting entire fields, journals, or conferences. *Data Representation:* Abstracts and titles often fail to fully represent the content of research papers, introducing uncertainty into the embeddings and potentially affecting cluster quality. *Cluster Verification:* While manual inspection suggested that clusters were semantically meaningful, the lack of rigorous empirical confirmation limits confidence in their meaningfulness. *Model Bias:* Clustering algorithms inherently introduce bias, and no single model is perfectly suited for every dataset.

3) External Validity: Temporal Bias: Sampling papers within specific timeframes risks introducing recency bias or excluding recent publications. *Replicability:* While the study demonstrated high replicability, applying these methods to larger or more diverse datasets may yield varying results depending on dataset composition.

B. Summary

This paper demonstrates that meta-research analysis can be effectively conducted by embedding research abstracts into a vector space and clustering the resulting embeddings. This methodology enabled the identification of distinct clusters, validated through silhouette scores, which showed that K-means and Gaussian Mixture algorithms significantly outperformed other methods in producing well-separated clusters.

Although the focus of this study was limited to Software Engineering research—a relatively small subfield of Computer Science—the proposed techniques are scalable and adaptable to larger datasets across diverse disciplines. These tools have practical applications for researchers, students, and professionals seeking to explore the research landscape within specific domains. The approach offers a data-driven way to identify key areas, subfields, and trends in the literature.

Despite its contributions, the study also revealed areas for future improvement. Custom clustering methods tailored to high-dimensional, noisy, and overlapping clusters could enhance results. Additionally, integrating advanced semantic analysis could empirically validate the semantic coherence of clusters. Expanding this work to broader datasets or larger fields may further generalize the proposed techniques.

In conclusion, this study provides a foundation for improving clustering performance in meta-research and extracting meaningful patterns from large-scale textual datasets. These findings open new opportunities for advancing the understanding of research landscapes and enhancing data-driven exploration in scientific fields.

References

[1] F. Almeida and G. Xexéo, "Word embeddings: A survey," 2023.

[2] E. Rudkowsky, M. Haselmayer, M. Wastian, M. Jenny, Š. Emrich, and M. Sedlmair, "More than bags of words: Sentiment analysis with word embeddings," *Communication Methods and Measures*, vol. 12, no. 2-3, pp. 140–157, 2018.

[3] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *International journal of machine learning and cybernetics*, vol. 1, pp. 43–52, 2010.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[5] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," vol. 14, pp. 1532–1543, 01 2014.

[6] H. Liu, Y. Wu, and Y. Yang, "Analogical inference for multi-relational embeddings," in *International conference on machine learning*, pp. 2168–2178, PMLR, 2017.

[7] B. Ghojogh and A. Ghodsi, "Attention mechanism, transformers, bert, and gpt: tutorial and survey," 2020.

[8] F. Incitti, F. Urli, and L. Snidaro, "Beyond word embeddings: A survey," *Information Fusion*, vol. 89, pp. 418–436, 2023.

[9] A. Ghosal, A. Nandy, A. K. Das, S. Goswami, and M. Panday, "A short review on different clustering techniques and their applications," *Emerging Technology in Modelling and Graphics: Proceedings of IEM Graph 2018*, pp. 69–83, 2020.

[10] J. Camacho-Collados and M. T. Pilehvar, "From word to sense embeddings: A survey on vector representations of meaning," *Journal of Artificial Intelligence Research*, vol. 63, pp. 743–788, 2018.

[11] J. Ravi and S. Kulkarni, "Text embedding techniques for efficient clustering of twitter data," *Evolutionary Intelligence*, vol. 16, no. 5, pp. 1667–1677, 2023.

[12] A. Petukhova, J. P. Matos-Carvalho, and N. Fachada, "Text clustering with llm embeddings," *arXiv preprint arXiv:2403.15112*, 2024.

[13] F. Ebrahimi, M. Tushev, and A. Mahmoud, "Classifying mobile applications using word embeddings," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 31, no. 2, pp. 1–30, 2021.

[14] Qdrant, "Fastembed: Fast, accurate, lightweight python library for embedding generation." GitHub. Available: https://github.com/qdrant/fastembed. Accessed: Dec. 5, 2024.

[15] V. Raunak, V. Gupta, and F. Metze, "Effective dimensionality reduction for word embeddings," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)* (I. Augenstein, S. Gella, S. Ruder, K. Kann, B. Can, J. Welbl, A. Conneau, X. Ren, and M. Rei, eds.), (Florence, Italy), pp. 235–243, Association for Computational Linguistics, Aug. 2019.