Optimizing Document Clustering through Correlation-Driven Cluster Formation

Suraj Kashyap[1] and Uttam Mahata[1]

[1] Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Shibpur, Howrah, India

ACKNOWLEDGEMENT

I extend my sincere gratitude to Dr. Asit Kumar Das for his invaluable guidance and unwavering support throughout the research and writing of this paper. Dr. Asit Kumar Das's expertise, encouragement, and insightful feedback have played a pivotal role in shaping the methodology and analysis presented herein.

His commitment to academic excellence, innovative thinking, and dedication to fostering a conducive research environment have been instrumental in the development of this work. I am truly fortunate to have had the opportunity to benefit from Dr. Asit Kumar Das's mentorship, which has significantly enriched the quality and depth of this research.

I express my heartfelt thanks to Dr. Asit Kumar Das for his continuous encouragement, scholarly insights, and for being a beacon of inspiration in the pursuit of knowledge and research excellence.

Suraj Kashyap, Uttam Mahata

2nd Year Under Graduate Student

Indian Institute of Engineering Science and Technology, Shibpur

INDEX

Optimizing Document Clustering through Correlation-Driven Cluster Formation

Suraj Kashyap[1] and Uttam Mahata[1]

[1] Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Shibpur, Howrah, India

ABSTRACT

Document clustering is a vital component in organizing extensive datasets for efficient information retrieval and analysis. This paper introduces an innovative approach to enhance document clustering accuracy through correlation-driven optimization. Our methodology, anchored in a dynamic correlation-based thresholding mechanism, results in clusters with adaptive sizes, reducing errors and showcasing the efficacy of the proposed approach. Comparative evaluations against traditional clustering algorithms, including K-Means, Affinity Propagation, Gaussian Mixture, and Agglomerative Clustering, underscore the superiority of our correlation-driven optimization in achieving improved clustering performance.

KEYWORDS

Information Retrieval, Threshold Value, Spearman Correlation Coefficient, Jaccard Coefficient, Cluster Formation, Error Calculation, Elbow Method, Comparative Analysis, K-Means Algorithm, Affinity Propagation, Gaussian Mixture Model, Agglomerative Clustering, Silhouette Score, Davies-Bouldin Index, Calinski Harabasz Score, Adjusted Rand Score, Normalized Mutual Information Score

1. INTRODUCTION

In the dynamic realm of digital content analysis, the critical task of document clustering stands as a pivotal step for uncovering underlying patterns. Traditional methodologies have laid a solid foundation, yet the challenge persists in achieving a harmonious balance between cluster granularity and adaptability to diverse document relationships. This paper introduces an innovative approach to document clustering optimization through correlation-driven techniques.

Our methodology is centered on the fundamental task of creating clusters from a given dataset using a carefully selected threshold value. Termed as 'adaptive sizes,' this threshold-driven approach enables dynamic adjustments to cluster sizes, enhancing the algorithm's flexibility. At the core of our strategy is the integration of correlation-based thresholding, strategically applied to mitigate errors related to misclassifications and overlap during the cluster formation stage.

This paper delves into the intricacies of our clustering methodology, detailing the experimental setups and presenting a comparative analysis against established clustering algorithms, including K-Means, Affinity Propagation, Gaussian Mixture, and Agglomerative Clustering. The results not only showcase the efficacy of our correlation-driven optimization approach but also offer insights into its performance across diverse datasets, highlighting its merits in the context of document clustering tasks.

2. LITERATURE REVIEW

2.1 Traditional Document Clustering Techniques

Document clustering, a fundamental task in information retrieval and text mining, has witnessed significant development over the years. Traditional techniques, such as K-Means, Affinity Propagation, Gaussian Mixture, and Agglomerative Clustering, have formed the basis for many document clustering approaches.

K-Means Clustering: One of the most widely used methods, K-Means partitions documents into 'k' clusters based on feature similarity. However, K-Means has limitations, especially regarding sensitivity to initial centroids and a predefined number of clusters.

Affinity Propagation: This clustering algorithm identifies exemplars among data points and forms clusters based on message-passing between data points. It is effective but may lead to an overestimation of clusters.

Gaussian Mixture Model (GMM): GMM assumes that the data is generated from a mixture of several Gaussian distributions. It is particularly useful for capturing complex data patterns but may struggle with non-Gaussian distributions.

Agglomerative Clustering: This hierarchical clustering method iteratively merges clusters based on a linkage criterion. While intuitive, it can be computationally expensive for large datasets.

These clustering algorithms serve as benchmarks for evaluating the Correlation Based Adaptive Clustering algorithm's performance. By comparing the results across these diverse algorithms, we gain insights into the strengths and limitations of each approach and assess the adaptability of the proposed algorithm across different clustering scenarios.

2.2 Challenges in Traditional Approaches

Traditional approaches face challenges in adapting to the dynamic relationships among documents, often leading to suboptimal cluster assignments. As datasets grow in complexity and size, there is an increasing need for more adaptive and error-resilient clustering techniques.

2.3 Correlation-Driven Optimization

This paper proposes a novel approach to document clustering optimization based on correlation-driven techniques, focusing on refining cluster assignments and introducing adaptive sizing to overcome challenges posed by traditional techniques.

2.4 Significance of Adaptive Cluster Sizes

The adaptive sizing of clusters, a distinctive feature of our methodology, adds a layer of flexibility to the clustering process. Traditional methods often struggle with fixed cluster sizes, leading to suboptimal results when handling documents with varying degrees of relatedness.

2.5 Comparative Analysis

This paper contributes to the literature by presenting a comparative analysis of our correlation-driven optimization against traditional clustering algorithms. The subsequent sections delve into the experimental setups and outcomes, showcasing the efficacy of our proposed approach in overcoming challenges posed by traditional techniques.

3. METHODOLOGY

Threshold-based Correlation Clustering (TBCC) is designed to optimize clusters based on the correlation matrix of TF-IDF vectors and a given threshold value 'T'. This method capitalizes on the semantic relationships between words to form cohesive and meaningful clusters. The process involves several key steps:

3.1 Correlation Matrix Calculation: The procedure commences with the calculation of the correlation matrix (C) derived from TF-IDF vectors. Spearman correlation is employed to robustly measure monotonic relationships between words, ensuring a comprehensive understanding of semantic associations.

$$C_{ij} = \text{Spearman\_Corr}(TF - IDF_i, TF - IDF_j)$$

This step guarantees that the correlation matrix captures both linear and potential nonlinear dependencies, enriching the semantic insights.

3.2 Creating Clusters: This step is designed to capture the semantic coherence among words by forming groups of words with above-average correlations.

- **Row-wise Calculation:** The row-wise calculation involves evaluating the average correlation for each word, providing a measure of its overall association with other words in the dataset.
- **Cluster Formation:** Clusters are then formed by selecting words whose individual column-wise average correlation values exceed the global average. This process ensures that each cluster comprises words with above-average semantic correlations.

$$\text{Cluster}_k = \{\text{word}_i \mid \text{Corr}_{ki} > \text{Average Corr}_k > T\}$$

This clustering strategy focuses on grouping words with stronger-than-average semantic relationships, setting the stage for subsequent optimization steps.

3.3 Jaccard Coefficient Calculation
The Jaccard coefficient is calculated between pairs of clusters, measuring the similarity by the size of their intersection divided by the size of their union.

$$\text{Jaccard Coefficient}(A, B) = \frac{A \cap B}{A \cup B}$$

This measure quantifies the degree of overlap between clusters, providing a meaningful metric to assess the distinctiveness and coherence of the formed semantic clusters.

3.4 Threshold based Cluster Merging: TBCC iteratively optimizes clusters by merging those with significant correlations above a specified threshold T. This iterative process ensures the formation of meaningful and distinct clusters.

i.e. Merge Cluster A and Cluster B if $J(A, B) \geq T$

This approach systematically refines clusters, producing semantically cohesive clusters representing distinct concepts or themes within the dataset.

3.5 Convergence Criteria: The optimization process continues until convergence, ensuring that no pair of clusters has a maximum correlation above the specified threshold value T.
If $\max(C_{ij}) < T$, the optimization process converges.

3.6 Cluster Refinement for Unassigned Columns
To refine the clustering results, an iterative procedure assesses columns initially unassigned to any clusters. The algorithm identifies clusters with a single column representation and evaluates the relationships between unassigned columns and those clusters.

Refinement Criteria: For each unassigned column, it assesses its relationship with columns in clusters with a single representation. A refinement criterion stipulates that if at least 75% of columns within the same original cluster support the assignment of an unassigned column, it is reassigned to that cluster.

Assignment Process: If the criteria are met, the unassigned column is reassigned to the identified cluster. If the criteria are not met, the column is treated as a singleton cluster.

3.7 Cluster Centroid Calculation: The centroid of each cluster is computed as the mean of TF-IDF vectors for all points within the cluster. This centroid represents the central theme or concept of the cluster.

$$Centroid_k = \frac{\sum_{i \in Cluster_k} TF - IDF_i}{Number\ of\ Documents\ in\ Cluster_k}$$

3.8 Error Calculation: The error calculation is pivotal in assessing clustering quality. Our approach employs the Root Mean Square (RMS) difference between document TF-IDF vectors and their assigned cluster centroid.

- **RMS Calculation**: $RMS(Document, Cluster) = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}}$
- **Cluster-wise Error**: Cluster-wise error ($E_{cluster}$)averages RMS values across all documents in a cluster.

  $E_{cluster} = \frac{\sum_{j=1}^{m} RMS(Document_j, cluster)}{m}$ where m is the number of documents in the cluster
- **Global Error**: Global error (E_{global}) is the average of cluster-wise errors across all clusters:

  $E_{global} = \frac{\sum_{k=1}^{K} E_{cluster_k}}{K}$ where K is the total number of clusters.

3.9 Final Clusters: The resulting clusters represent semantically related words, forming distinct semantic units. These clusters are the output of the TBCC algorithm and can be further analyzed for insights into the underlying semantic structure of the dataset.

The TBCC method, outlined above, provides a robust framework for semantic clustering, offering a unique and effective approach for discovering latent semantic structures within textual data. This methodology contributes to the growing field of clustering techniques by leveraging correlations and semantic relationships among words.

4. EXPERIMENTAL SETUP

4.1 DATASET DESCRIPTION

The dataset used in our experimentation consists of 200 questions distributed across four distinct topics, each comprising 50 questions. The chosen topics are Biotechnology, Database Management System (DBMS), Network and Networking, and Climate Change. It is important to note that these topics were selected arbitrarily, devoid of any specific influence on the functionality or performance of our custom clustering algorithm.
In initial phase, our objective is to transform raw text data into a format suitable for clustering. This involves a series of essential steps: **Tokenization, Lemmatization, Filtering stopwords and punctuation, TF-IDF Vectorization**. The preprocessed text data is then transformed into numerical vectors using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique. This dataset forms the basis for assessing the clustering algorithm's performance and comparing it against established methods.

4.2 Parameter Settings

In our experiment, we focused on determining the optimum threshold value for our custom clustering algorithm. The threshold value plays a crucial role in defining the strength of semantic correlations between words, thereby influencing the formation of clusters. To thoroughly explore the threshold space, we varied the threshold values in the range of 0.2 to 0.3, incrementing by 0.01 for each iteration. For each threshold value, we calculated the corresponding number of clusters and the associated error. The process involved the following steps:

- **Threshold Variation**: Threshold values were systematically varied from 0.2 to 0.3. For each threshold value, the custom clustering algorithm was applied to the dataset, resulting in a unique clustering arrangement.
- **Elbow Method for Optimum Threshold**: The elbow method involves plotting the rate of change in error against different threshold values. The point at which the rate of error reduction significantly slows down resembles the bending point of an elbow in the plot. We identified the threshold value corresponding to this elbow point as the optimum threshold, balancing cluster coherence (minimum error) and meaningfulness (reasonable number of clusters).
- **Optimum Threshold Identification**: The optimum threshold value, determined using the elbow method, was selected for subsequent analysis.

After determining the optimum threshold value, we set this value for subsequent analyses, ensuring consistent and comparable results. The fixed threshold value provides stability to the clustering process, allowing for a more focused evaluation of the algorithm's performance.

Next, we employed two different scenarios for comparison:

- **Comparison at Optimum Number of Clusters**: The custom algorithm's performance was assessed against other clustering algorithms at the optimum number of clusters, determined by the identified threshold value.

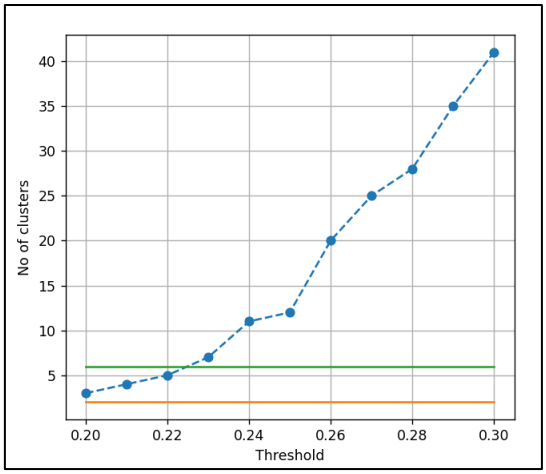
- **Comparison at Original Number of Clusters**: The number of clusters was set to four, reflecting the original categorization based on the four chosen topics (Biotechnology, DBMS, Network and Networking, and Climate Change).

The custom algorithm's results were compared with other algorithms under the constraint of the optimum number of clusters and original number of clusters.

This parameterization strategy allowed us to not only optimize the algorithm's performance under its intrinsic settings but also evaluate its adaptability to specific scenarios, contributing to a comprehensive understanding of its capabilities.

5. RESULTS

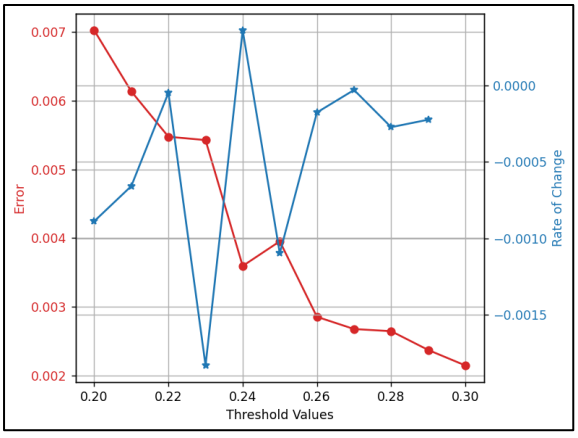
5.1 No of Clusters vs Threshold Value: Graph 1 illustrates the relationship between the number of clusters and the threshold value. The graph showcases the variation in the number of clusters as the threshold value changes.



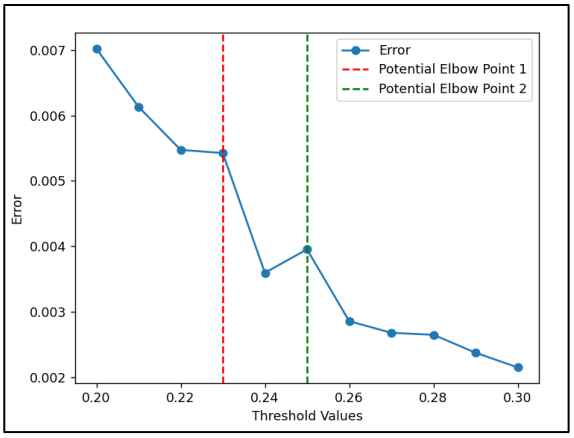
Graph 1

5.2 Error and Rate of Change vs Threshold Value: Graph 2 displays the Error and Rate of Change in Error on the same graph, providing insights into the algorithm's performance across different threshold values.

5.3 Optimum Threshold Value Identification: The elbow method is employed to determine the optimum threshold value. Graph 3 visually represents the identification of the optimum value based on the rate of change in error.



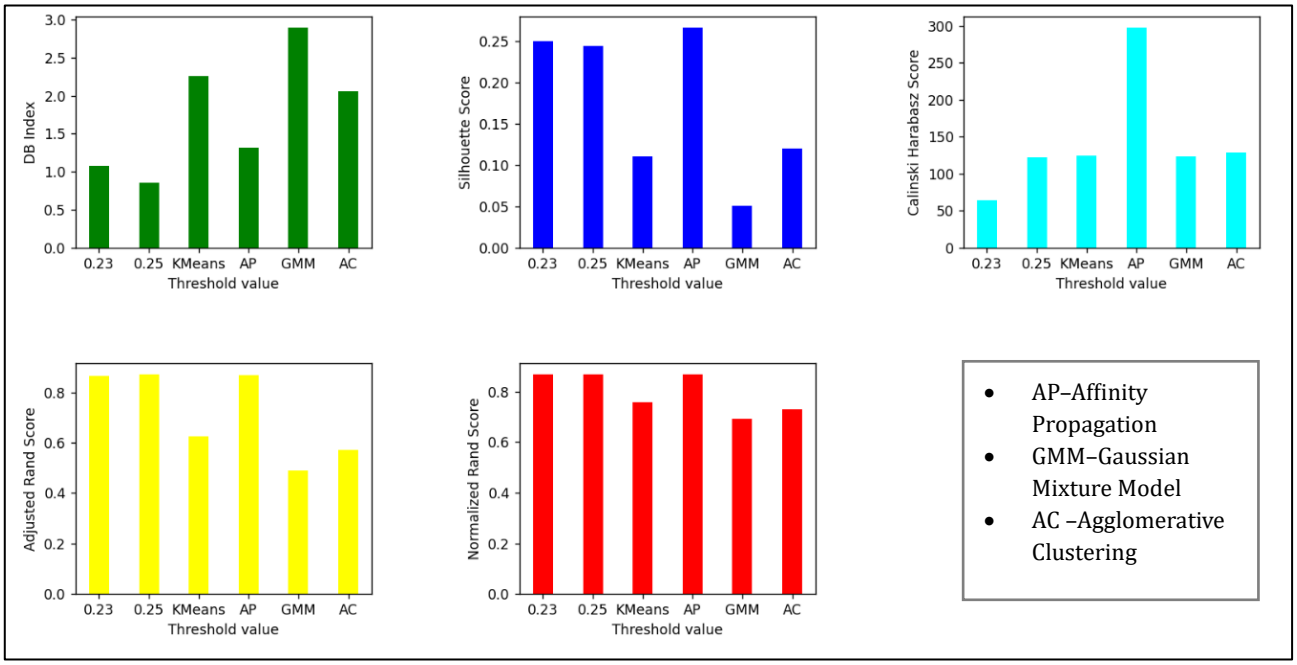
Graph 2



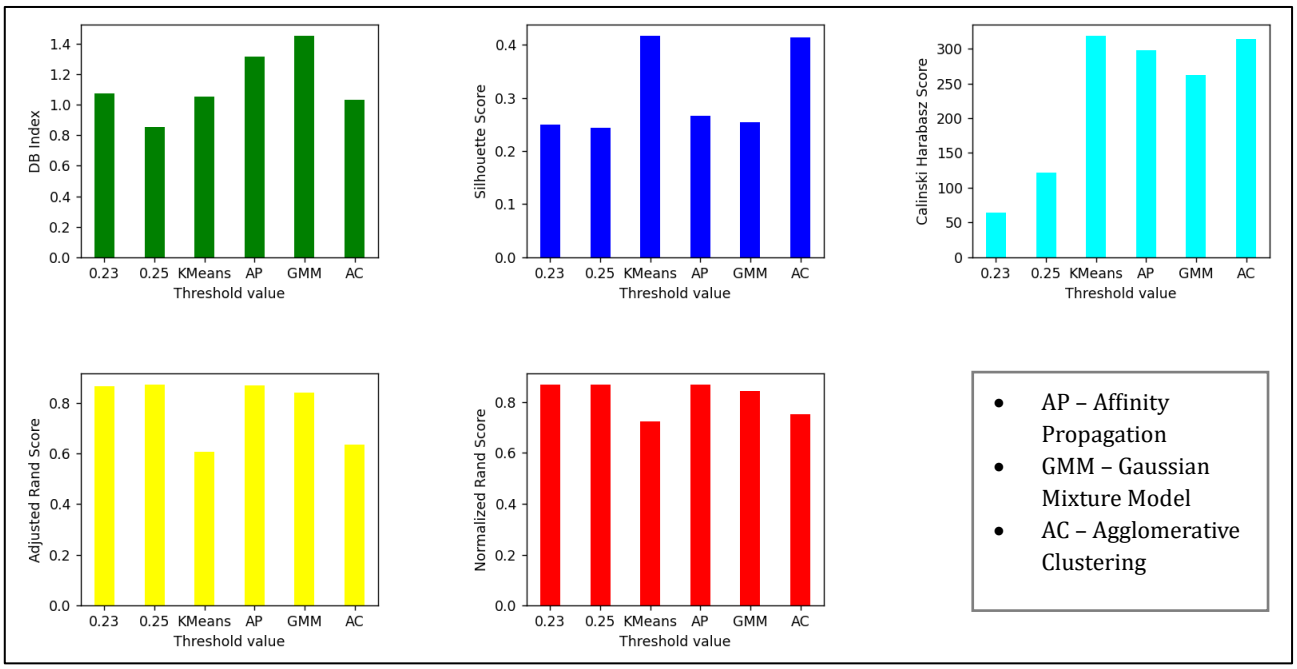
Graph 3

5.4 Computational Efficiency: The custom clustering algorithm demonstrates efficient performance, with clustering, centroid and error calculation taking approximately 3-4 seconds.

5.5 Comparative Analysis: Comparisons between the optimum threshold values (0.23 and 0.25) and other algorithms are conducted. The algorithm's results are evaluated against other standard clustering algorithms under the same (Graph 4) and original number of clusters i.e, 4 (Graph 5).



Graph 4



Graph 5

6. DISCUSSION

6.1 Interpretation of Results: The results obtained from the experiments provide valuable insights into the performance of the proposed Custom Clustering Algorithm compared to traditional clustering approaches. Two key scenarios were considered for evaluation: clustering using original number of clusters (4) and number of clusters corresponding to the optimum threshold value (12).

6.1.1 Optimum Number of Clusters (12)

When determining the optimum number of clusters using the elbow method, the algorithm displayed superior performance with threshold values of 0.23 and 0.25 compared to standard clustering methods. Despite the inherent challenge of setting an optimal threshold, the algorithm showcased better Silhouette scores, Calinski Harabasz scores, and Adjusted Rand Scores. The Normalized Mutual Information Score also demonstrated the algorithm's effectiveness in capturing meaningful semantic relationships.

Model	Davies-Bouldin Index	Silhouette Score	Calinski Harabasz Score	Adjusted Rand Score	Normalized Mutual Info Score
Custom Clustering Algorithm (0.23)	1.0755	0.2502	63.9523	0.8651	0.8684
Custom Clustering Algorithm (0.25)	0.8524	0.2445	122.0241	0.8721	0.8694
KMeans	2.2553	0.1106	124.9375	0.6236	0.7595
Affinity Propagation	1.3178	0.2664	297.9176	0.8666	0.8675
Gaussian Mixture	2.8988	0.0506	122.9254	0.4891	0.6936
Agglomerative Clustering	2.0598	0.1198	128.7157	0.5726	0.7294

6.1.2 Original Number of Clusters (4)

The Custom Clustering Algorithm, with threshold values of 0.23 and 0.25, demonstrated competitive performance compared to standard clustering algorithms such as KMeans, Affinity Propagation, Gaussian Mixture, and Agglomerative Clustering. The Davies-Bouldin index, Silhouette score, Calinski Harabasz score, Adjusted Rand Score, and Normalized Mutual Information Score were used as metrics for evaluation.

Model	Davies-Bouldin index	Silhouette Score	Calinski Harabasz score	Adjusted Rand Score	Normalized Mutual Info Score
Custom Clustering Algorithm (0.23)	1.0755	0.2502	63.9523	0.8651	0.8684
Custom Clustering Algorithm (0.25)	0.8524	0.2445	122.0241	0.8721	0.8694
KMeans	1.0535	0.4173	318.6811	0.6050	0.7224
Affinity Propagation	1.3178	0.2664	297.9176	0.8666	0.8675
Gaussian Mixture	1.4542	0.2545	261.868	0.8390	0.8438
Agglomerative Clustering	1.0346	0.4144	314.3739	0.6346	0.7512

The results indicate that the proposed algorithm achieved lower Davies-Bouldin indices, higher Silhouette scores, and competitive Calinski Harabasz scores. The Adjusted Rand Score and Normalized Mutual Information Score also highlight the algorithm's ability to form clusters with meaningful semantic associations.

6.2 Algorithm's Strengths and Weaknesses

The algorithm's strengths lie in its adaptability to varying document relationships through dynamic correlation-based optimization. By leveraging semantic relationships, the algorithm creates cohesive and meaningful clusters. However, a potential limitation is the sensitivity to the choice of the correlation threshold, which warrants further exploration.

6.3 Potential Applications

The Custom Clustering Algorithm holds promise in various applications where document organization and extraction of semantic patterns are crucial. Its ability to dynamically adjust to document relationships makes it suitable for datasets with diverse content and varying semantic associations.

6.4 Future Work

Future research could explore mechanisms for automated selection of correlation thresholds based on the dataset's characteristics. Additionally, scalability and robustness assessments on larger and more diverse datasets could further validate the algorithm's effectiveness.

Appendix A: Pseudo Code for Custom Clustering Algorithm

The following pseudo code outlines the key steps of the custom clustering algorithm. Detailed descriptions of each function and step can be found in the respective sections of the methodology.

```
# Step 1: Threshold-based Correlation Clustering
function threshold_based_correlation_clustering(data):
    # 1.1 Correlation Matrix Calculation
    correlation_matrix = calculate_correlation_matrix(data)
    # 1.2 Creating Clusters
    clusters = create_clusters(correlation_matrix)
    # 1.3 Jaccard Coefficient Calculation
    jaccard_coefficients = calculate_jaccard_coefficients(clusters)
    # 1.4 Threshold-based Cluster Merging
    merged_clusters = threshold_based_cluster_merging(clusters, jaccard_coefficients)
    # 1.5 Convergence Criteria
    while not convergence_criteria(merged_clusters):
        # Repeat steps 1.3 to 1.4 until convergence
    return merged_clusters

# Step 2: Recalculate Clusters for Unassigned Columns
function recalculate_clusters(data, original_clusters):
    labels = initialize_labels(data)
    for label in original_clusters:
        if cluster_size(label) == 1:
            recalculate_label(data, labels, label, original_clusters)
    return labels

# Step 3: Optimization Process
function optimization_process(data):
    labels = threshold_based_correlation_clustering(data)
    # 3.1 Error Calculation
    error = calculate_error(data, labels)
    # 3.2 Centroid Identification
    centroids = identify_centroids(data, labels)
    return labels, error, centroids

# Main Function
function main(data):
    # 1. Custom Clustering Algorithm
    labels, error, centroids = optimization_process(data)
    # 2. Recalculate Clusters for Unassigned Columns
    labels = recalculate_clusters(data, labels)

# Execute the main function with your dataset
main(your_dataset)
```

References

[1] R.Jensi, G.Wiselin Jiji, C. C. "A Survey on Optimization Approaches to Text Document Clustering." International Journal on Computational Science & Applications (2014): Doi:10.5121/ijcsa.2013.3604.

[2] Asit Kumar Das (2011). Ph.D. (Engineering), Department of Computer Science and Technology, Bengal Engineering and Science University.