Journal of Computer Science
https://dergipark.org.tr/en/pub/bbd
Anatolian Science

ISSN,e-ISSN: 2548-1304
Volume:IDAP-2023, pp:90-99, 2023
https://doi.org/10.53070/bbd.1350971

Research Paper

Development of Text Summarization Method based on Graph Theory and Malatya Centrality Algorithm

Cevher Tayyib BAKAN*[1], Selman YAKUT[1]

[1]Software Engineer Department, Inonu University, Malatya, Turkey

(cevherbakan@gmail.com, selman.yakut@inonu.edu.tr)

Received:Aug.27,2023 Accepted:Oct.17,2023 Published:Oct.18,2023

Abstract— With the advancement of the internet, humanity has gained easy access to a plethora of information. However, to access accurate content, numerous texts and sources must be read. These texts often contain repetitive words and sentences. The abundance of information renders reading texts in their entirety inefficient in terms of time and makes finding suitable content challenging. To overcome these difficulties, various methods have been developed in research on automatic summarization. In the literature, there are numerous methods developed for different purposes in text summarization. Nevertheless, text summarization can generally be divided into two distinct categories: extractive and abstractive summarization. Abstractive algorithms tend to create new sentences by learning from the text. However, this approach prolongs the working process due to the learning phase and the generated sentences may not possess absolute accuracy. On the other hand, extractive methods, if unable to generate new sentences, have the ability to provide faster and completely accurate summaries by selecting sentences that already exist in the text. For these reasons, in our study, the aim is to perform text summarization using graph theory and the Malatya Centrality Algorithm. The Malatya Centrality Algorithm offers a polynomial approach to solving Vertex Cover Problems and is regarded as an effective solution method. It is believed that the Malatya Centrality Algorithm will contribute to graph-based text summarization. The implementation has been developed using the Python programming language, and the obtained results have been evaluated.

Keywords: *Graph-Based Text Summarization, Malatya Centrality Algorithm, Text Summarization*

1. Introduction

Text summarization is a widely used method to alleviate information overload, present lengthy texts concisely, or emphasize the main points of a text. However, traditional text summarization methods can sometimes encounter limitations and yield challenging results to obtain accuracy. With the advancement of internet technology, humanity has rapidly gained access to an abundance of information sources. Nonetheless, this wealth of information has made accessing accurate and relevant content cumbersome, rendering reading texts inefficient in terms of time. Frequently recurring words and sentences within texts have posed an obstacle to accessing essential information. In order to address this issue and provide an effective text summarization method, research has been conducted on automatic summarization, leading to the development of various techniques.

In text summarization studies, two main approaches are commonly employed: extractive summarization and abstractive summarization. In extractive summarization, the primary objective is to identify the most important sentences. This involves detecting word repetitions and similar phrases to determine the significance of sentences. On the other hand, abstractive summarization involves extracting meaningful new sentences from a given text using methods such as deep learning and artificial neural networks. Within the scope of the literature examined in this article, numerous studies have been conducted on various methods and algorithms in the field of text summarization. Notably, graph-based extractive approaches and machine learning-based abstractive methods have garnered attention. For instance, a study highlighted the potential contributions of the Malatya Centrality Algorithm in solving Vertex Cover Problems through a polynomial approach and its potential implications for text summarization (Yakut S, Oztemiz F, Karci A, 2022). In a study on text summarization, a new approach for single-document text summarization has been proposed, where texts are represented using graphs, sentences are weighted, and the suggested method has shown competitive performance compared to traditional methods (Hark C, Taner Uçkan T, Seyyarer E., Karcı A, 2019). In another study on text summarization, a new method has been introduced. In this method, it's observed that overlap and repetition can be simultaneously optimized, which significantly enhances the summary quality in terms of ROUGE metrics (Hark C, Taner Uçkan T, Karcı A, 2022). Similarly, other researchers have developed different approaches using graph-based ranking algorithms, genetic algorithms, deep learning techniques, and other methods for text summarization.

In this study, the aim is to develop a text summarization method based on graph theory and the Malatya Centrality Algorithm. Given the polynomial approach of the Malatya Centrality Algorithm to graph-based extractive methods, it is believed to possess the potential to generate faster and more comprehensible sentences. The proposed method seeks to expedite the transformation of the text into a summary by identifying crucial nodes within the text, aiming for more effective results. This work intends to contribute a novel graph-based extractive approach to the field of text summarization, offering a fresh perspective to the literature. The utilization of the Malatya Centrality Algorithm accelerates the text summarization process and yields efficient outcomes. The algorithm introduces a new approach to solving the Minimum Vertex Cover Problem and has been demonstrated to be successful in previous studies. Moreover, it presents a new approach to reevaluating the concept of centrality on graphs.

2. Related Studies in the Literature

The theme of text summarization encompasses a broad scope within the realm of academic literature. In this context, certain studies utilize graph-based inference methods, while others opt for interpretive approaches such as machine learning. Within the domain of inference-focused methods, we have scrutinized exemplar studies in the literature. For instance, Tülek employed stemming algorithms and root and affix analysis to generate summaries of Turkish texts, subsequently identifying sentences to be included in the summary (Tülek, M., 2007). On the other hand, Khushboo et al. amalgamated graphical ranking algorithms with shortest path algorithms to enhance automatic sentence extraction techniques (Khushboo S. Thakkar, R.V. Dharaskar, & M.B. Chandak, 2010). Erkan and Dragomi, in their study, embraced the LexRank approach, evaluating sentence importance by adopting the principle of eigenvector centrality through graphical representation of sentences. This model involves constructing a connection matrix based on cosine similarity within the graphical representation of sentences (Güneş Erkan, & Dragomir R. Radev, 2004). Moawad and Aref adopted an approach wherein they represented the original document using a rich semantic graph to develop a summarization strategy. Herein, the generated graph is simplified to create a more concise document (Ibrahim F. Moawad, & Mostafa Aref, 2013). Ferreira et al. introduced an innovative graphical model for text processing applications, based on four distinct dimensions: similarity, semantic similarity, common references, and discourse information (Rafael Ferreira, Frederico Freitas, Luciano de Souza Cabral, Rafael Dueire Lins, Rinaldo Lima, Gabriel França, Steven J. Simskez, & Luciano Favaro, 2013).

Mallick et al. proposed a graph-based text summarization method that captures the essence of a text document. This approach, developed by leveraging a modified TextRank calculated based on the PageRank principle tailored to each webpage, provides an approximation of the content (Chirantana Mallick, Ajit Kumar Das, Madhurima Dutta, Asit Kumar Das, & Apurba Sarkar, 2018). Sankarasubramaniam et al. meticulously examined a novel approach by amalgamating Wikipedia with graph-based ranking. This methodology involves utilizing a pre-constructed two-part sentence-concept graph, and subsequently applying iterative updates to this graph to arrange introductory sentences (Yogesh Sankarasubramaniam, Krishnan Ramanathan, & Subhankar Ghosh, 2014). In their study, Alguliev and Aliguliyev investigated an unsupervised document summarization method, exploring techniques for clustering and extracting sentences from the original document to form a summary. They also introduced new criterion functions for sentence clustering as part of their research (Rasim ALGULIEV, & Ramiz ALIGULIYEV, 2009).

The study by Nagwani and Verma demonstrates the design and implementation of a text summarization algorithm based on frequent terms in the Java programming language. The developed method consists of three main steps. In the initial step, stopwords are eliminated from the document to be summarized, and stemming is applied to convert words into their root forms. In the second step, term-frequency data is calculated from the document, and frequently used terms are selected; semantically equivalent terms are generated for these selected terms. Finally, in the third step, sentences containing frequently occurring and semantically equivalent terms are filtered to create the summary (Naresh Kumar Nagwani, & Dr. Shrish Verma, 2011).

In Mihalcea's study, a method utilizing graph-based ranking algorithms is presented for automatic sentence extraction (Rada Mihalcea, 2004). Akter and fellow researchers have proposed a text summarization method that extracts significant sentences from single or multiple Bengali documents. In this approach, input documents undergo preprocessing steps such as tokenization and stemming; subsequently, Term Frequency-Inverse Document Frequency (TF/IDF) is used to calculate word scores, and the scores of these words are aggregated to determine sentence scores. Additionally, the study employs the K-means clustering algorithm (Sumya Akter et al., 2017).

Other studies in the literature also demonstrate that sentiment-based approaches have garnered significant interest. Ježek and Steinberger have provided a comprehensive overview of summarization methods, encompassing various types from classical techniques to compilation-based and information-rich approaches (Karel Ježek, & Josef Steinberger, 2007). Babar and Patil's work focuses on semantic approaches such as Fuzzy Logic Inference and Latent Semantic Analysis for text summarization (S.A. Babar, & Pallavi D. Patil, 2015). Additionally, Ozsoy et al. have introduced distinct Latent Semantic Analysis (LSA) based summarization algorithms, with these proposals originating from the authors of the article as well (Makbule Gulcin Ozsoy, & Ferda Nur Alpaslan, 2011).

Some studies that employ techniques like machine learning and deep learning have also been examined. Erhandı conducted a summarization study on Turkish and English texts using the deep learning approach (Erhandı B., 2020). Neto et al. presented a summarization process for text using trainable machine learning algorithms, which utilize features extracted directly from the original text (Joel Larocca Neto, Alex A. Freitas, & Celso A. A. Kaestner, 2003). Silla et al. tackled text summarization as a classification problem and developed an automatic summarization method. In this approach, summaries for documents were generated based on attributes that define the documents. The aim of the study is to investigate the effectiveness of Genetic Algorithm-based feature selection in improving the performance of classification algorithms (Carlos N. Silla Jr., Gisele L. Pappa, Alex A. Freitas, & Celso A. A. Kaestner, 2004).

Several studies developed using genetic algorithms have been discussed. In the study by Kaynar, a genetic algorithm was employed for sentence-based summarization from text. The genetic algorithm was utilized to train the system using datasets (O. Kaynar, Y. E. IŞIK, Y. GÖRMEZ, & F. DEMİRKOPARAN, 2017). Al-Abdallah and Al-Taani proposed the Particle Swarm Optimization (PSO) algorithm for Arabic document summarization in their research. The PSO approach was compared with Evolutionary Algorithms (EA) and Harmony Search (HS) methods (R. Z. Al-Abdallah, & A. T. Al-Taani, 2017). Jain and his team suggested the use of Real-Coded Genetic Algorithm (RCGA) on health text data from the Kaggle dataset to develop an Automatic Text Summarization (ATS) methodology for Hindi. The proposed methodology encompasses preprocessing, feature extraction, processing, sentence ordering, and summary generation steps (A. Jain, A. Arora, J. Morato, D. Yadav, & K. V. Kumar, 2022).

3. Proposed Method

In extractive text summarization, graph-based methods are commonly employed, aiming to identify the most influential nodes within the graph. Various algorithms are utilized to find these influential nodes. Some of these algorithms include the PageRank algorithm, Closeness algorithm, and Eigen algorithm. In this study, however, the Malatya Centrality Algorithm is utilized to determine the most influential nodes. The Malatya Centrality Algorithm presents a polynomial remedy for addressing the Vertex Cover Dilemma, encompassing a duo of discrete phases: computing the Malatya centrality score and cherry-picking the pertinent nodes. The Malatya centrality score for every distinct node within the graph is established by the cumulative fusion of proportions, particularly the proportion of the node's degree to the collective sum of degrees among its adjacent nodes. The subsequent stage necessitates resolving a quandary concerning the selection of nodes, aiming to craft a vertex enclosure (Yakut S, Oztemiz F, Karci A, 2022).

In this study, text summarization was carried out using an approach based on the Malatya algorithm. The flowchart of the proposed method is presented in Figure 3.1. The pseudocode associated with this approach is provided in Algorithm 1. In the proposed method, the input text data is transformed into a graph format. Using the Malatya algorithm, the centrality values of nodes in this graph structure are determined. The nodes with the highest values are selected, and nodes are chosen in a way that maximizes coverage within the graph. These identified nodes constitute the nodes responsible for the summarization process of the input text. By selecting these nodes, sentences that can represent the input text are chosen. In this algorithm, the number of nodes capable of representing the text is determined based on the input text.

In Figure 3.1, the flowchart of the algorithm illustrates the process of determining the stop words in the text to be summarized. Subsequently, these stop words are extracted and transferred to the graph. After each sentence is transferred to the graph, the Malatya Centrality algorithm is applied to the nodes of the graph to select the most influential node. In each step, the most influential node is removed from the graph, and the process is repeated to find the most influential node in the new graph. Finally, these most influential nodes are combined to create the summarized text. The detailed operation of this new method and the utilization of the Malatya Centrality Algorithm in text summarization can be observed in the flowchart.

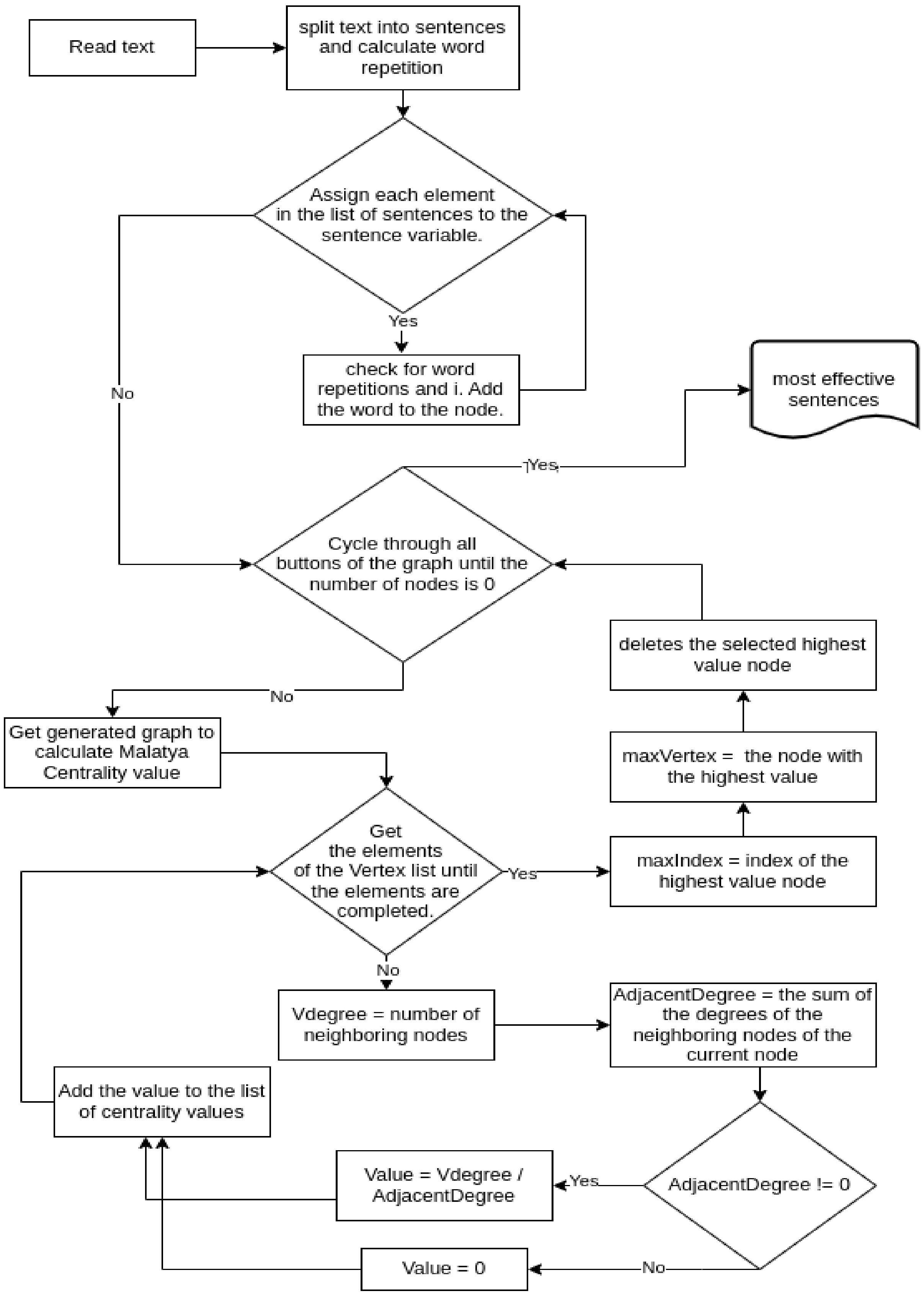
Read text
split text into sentences and calculate word repetition
Assign each element in the list of sentences to the sentence variable.
Yes
check for word repetitions and i. Add the word to the node.
No
most effective sentences
Yes
Cycle through all buttons of the graph until the number of nodes is 0
No
Get generated graph to calculate Malatya Centrality value
deletes the selected highest value node
maxVertex = the node with the highest value
maxIndex = index of the highest value node
Get the elements of the Vertex list until the elements are completed.
Yes
No
Vdegree = number of neighboring nodes
AdjacentDegree = the sum of the degrees of the neighboring nodes of the current node
Add the value to the list of centrality values
Value = Vdegree / AdjacentDegree
Yes
AdjacentDegree != 0
Value = 0
No

Figure 3.1. Flowchart of the Proposed Method

Proposed Algorithm Pseudocode

```
1.  function CalculateMalatyaCentrality(graph):
2.      vertexList = GetNodes(graph)  // Store nodes of the graph in an array
3.      centralityValues = []  // Initialize an empty list to store centrality values
4.      for each vertex in vertexList:
5.          vDegree = GetDegree(graph, vertex)
6.          adjacentDegree = 0  // Sum of degrees of neighbors of the current vertex
7.          for each neighbor in GetNeighbors(graph, vertex):
8.              adjacentDegree = adjacentDegree + GetDegree(graph, neighbor)
9.          if adjacentDegree ≠ 0:  // If the sum of neighbor degrees is not zero
10.             value = vDegree / adjacentDegree  // Calculate the centrality value
11.         else:  // If the sum of neighbor degrees is zero
12.             value = 0
13.         Add value to centralityValues
14.     return centralityValues
15. function FindMaxMalatyaCentralityValue(graph):
16.     centralityValues = CalculateMalatyaCentrality(graph)
17.     maxIndex = FindMaxIndex(centralityValues)  // Find the index of the maximum value
18.     maxVertex = GetNodeAtIndex(graph, maxIndex)  // Find the vertex associated
19.     RemoveNode(graph, maxVertex)  // Remove the vertex with the highest centrality value
20.     return maxVertex, GetEdgeCount(graph)
21. function FindMinVertexCover(graph):
22.     while GetEdgeCount(graph) ≠ 0:  // Repeat until there are no uncovered
    edges
23.         maxVertex, edgeCount = FindMaxMalatyaCentralityValue(graph)
24.     return maxVertex, edgeCount
25. function ReadText(url):
26.     return text
27. function CreateGraph(sentences, wordOccurrences):
28.     stopWords = GetStopwords()  // Get a list of common stopwords
29.     G = CreateEmptyGraph()  // Create an empty graph
30.     for each sentence in sentences:
31.         AddNode(G, sentence)  // Add a node for each sentence
32.         for each word, occurrenceCount in wordOccurrences:
33.             if WordExistsIn(sentence, word) and WordNotInStopwords(word):
34.                 AddEdge(G, sentence, word, weight=occurrenceCount)
35.     return G
36. function CalculateRouge(reference, summary):
37.     return  CalculateRougeMetrics(summary, reference)
38. function Main():
39.     text = ReadText()  // Read the text
40.     sentences = SplitSentences(text)  // Split the text into sentences
41.     wordOccurrences = CountWordOccurrences(text)  // Count occurrences of words
42.     G = CreateGraph(sentences, wordOccurrences)  // Create the graph
43.     minCover, edgeCount = FindMinVertexCover(G)  // Find the minimum vertex cover
44.     mostEffectiveSentences = GetSentenceByIndex(sentences, minCover)
45.     Print mostEffectiveSentences
46.     referenceText = "Sample Reference Text"  // Reference text for comparison
47.     scores = CalculateRouge(referenceText, mostEffectiveSentence)
48.     Print scores
```

4. Experimental Results

In extractive text summarization, graph-based methods are commonly employed. The primary objective of these methods is to identify the most influential nodes within the graph. Various algorithms are used to identify these influential nodes. In this study, the Malatya Centrality Algorithm will be utilized to determine the most effective nodes. The Malatya Centrality algorithm offers a polynomial approach to solving the Vertex Cover Problem. This approach consists of two steps: the calculation of the Malatya centrality value and the selection of covering nodes. The Malatya centrality value of each node within the graph is computed as the summation of ratios, specifically the ratio of the node's degree to the sum of degrees among its neighboring nodes. The second step involves addressing a node selection problem for the purpose of constructing a vertex cover.

To carry out this study, text summarization was performed using the Python programming language. The study allows text summarization both from files and web pages through URL input. For instance, when a portion of the text from the article "A New Approach Based on Centrality Value in Solving the Minimum Vertex Cover Problem" (Yakut S, Oztemiz F, Karci A, 2022) was tested, the text was processed as follows: A node was created for each sentence within the text, as illustrated in the diagram at each step. Subsequently, the words were added as edges to these nodes, and common words were identified as depicted in the graph.

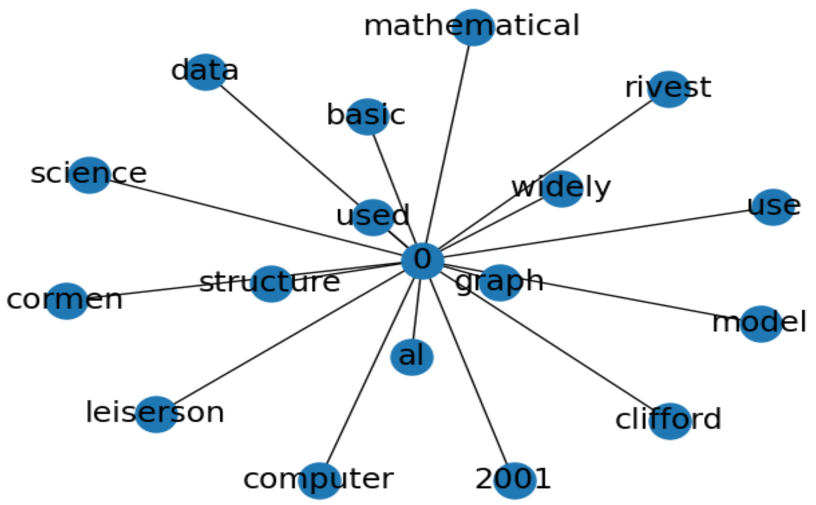

Figure 4.1. Graph Structure in the First Sentence

Figure 4.1 illustrates the addition of the first sentence to the graph, with each word being added as a node labeled as "0".

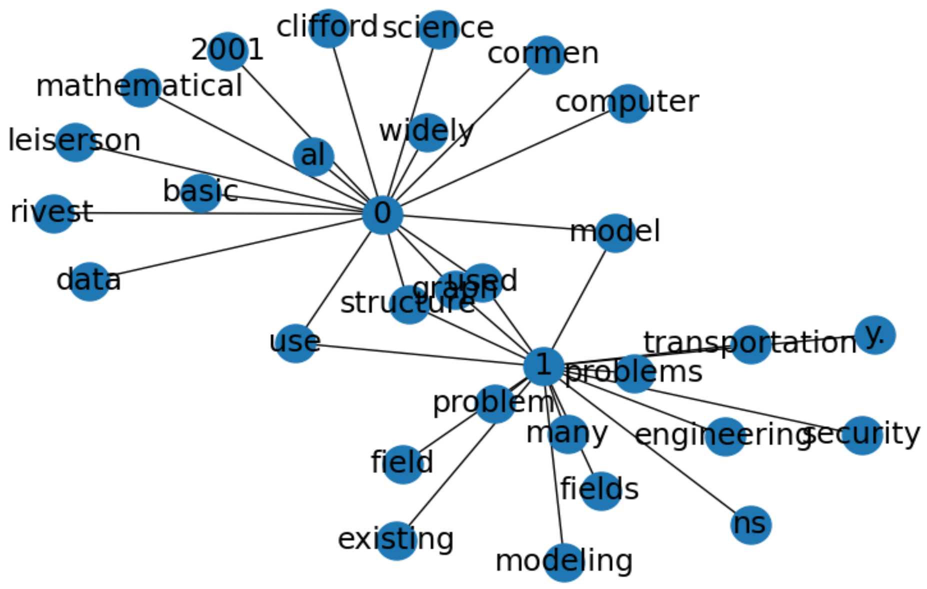

Figure 4.2. Depicts the graph structure after the addition of the second sentence

In Figure 4.2, upon adding the second sentence to the graph, the shared nodes in the first sentence are identified and included. This process reveals the relationship between the sentences.

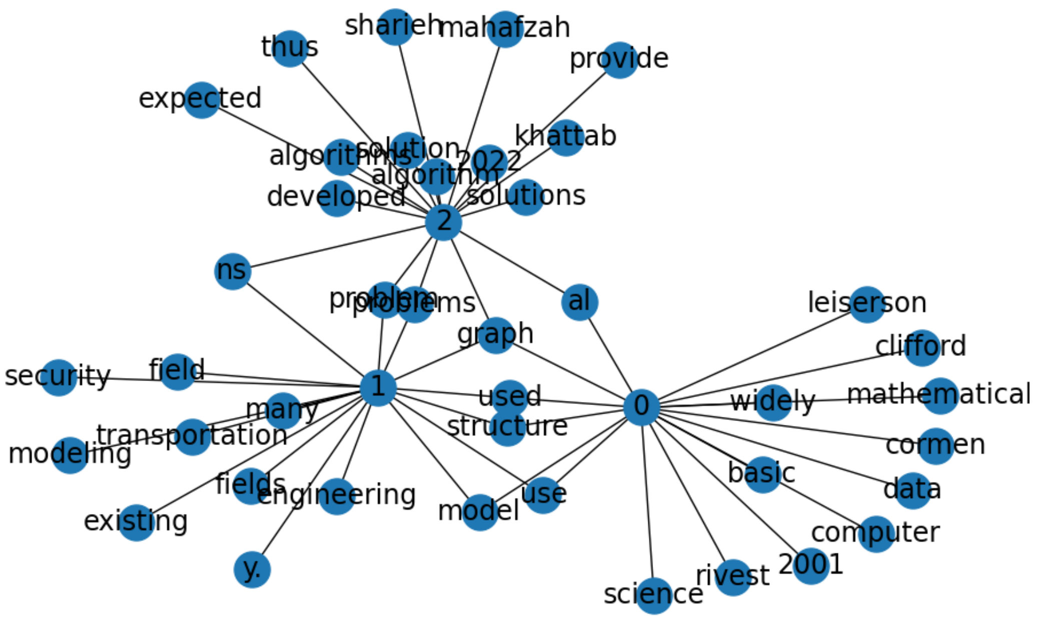

Figure 4.3. Graph Structure after Adding the 3rd Sentence

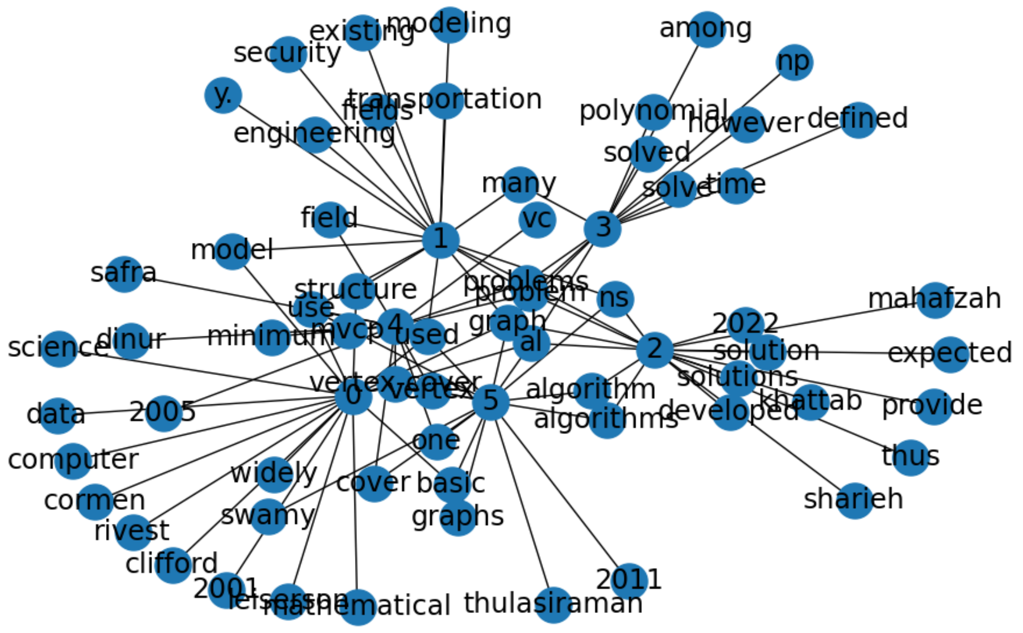

Figure 4.4. Graph Structure after Adding the 6th Sentence

In Figure 4.4, as we reach the 6th sentence in the graph, we can observe a further increase in the relationships between sentences. As we proceed, more common words will emerge, enabling us to select the most effective sentences from within the text.a

The same example text was tested with other algorithms used in text summarization. The compared algorithms to the proposed method are TextRank, LexRank, and PageRank algorithms. The Rouge scores of the results obtained from these algorithms were compared with the results generated from the proposed method. The comparisons were conducted based on Rouge-1 metric in Table-1, Rouge-2 metric in Table-2, and Rouge-l metric in Table-3.

Rouge-1 assesses the similarity of snippets of one-word length between the summarized text and the reference text. These scores measure the similarity at the word level. Rouge-2 measures the similarity of snippets with two-word length. Rouge-l utilizes a Longest Common Subsequence (LCS) based similarity measure, gauging the similarity of the longest shared subsequence between all summary and reference texts.

Table 1. Comparison of the Proposed Method with TextRank, LexRank, and PageRank Algorithms for Text Summarization based on ROUGE-1 Metric

Summarization Methods	Rouge-1		
	Sensitivity	Precision Ratio	F-Score
TextRank	0.32374	**0.51724**	0.39823
LexRank	0.39568	0.44354	0.41825
PageRank	0.28057	0.43333	0.34061
Proposed Method	**0.46762**	0.38690	**0.42345**

Table 2. Comparison of the Proposed Method with TextRank, LexRank, and PageRank Algorithms in Text Summarization based on ROUGE-2 Metric

Summarization Methods	Rouge-2		
	Sensitivity	Precision Ratio	F-Score
TextRank	0.11372	**0.21323**	**0.14833**
LexRank	0.09803	0.12886	0.11135
PageRank	0.07843	0.16129	0.10554
Proposed Method	**0.14117**	0.14173	0.14145

Table 3. Comparison of the Proposed Method with TextRank, LexRank, and PageRank Algorithms in Text Summarization based on ROUGE-L Metric

Summarization Methods	Rouge-l		
	Sensitivity	Precision Ratio	F-Score
TextRank	0.30215	**0.51724**	0.39823
LexRank	0.31654	0.44354	**0.41825**
PageRank	0.26618	0.43333	0.34061

Proposed Method	**0.41726**	0.34523	0.37785

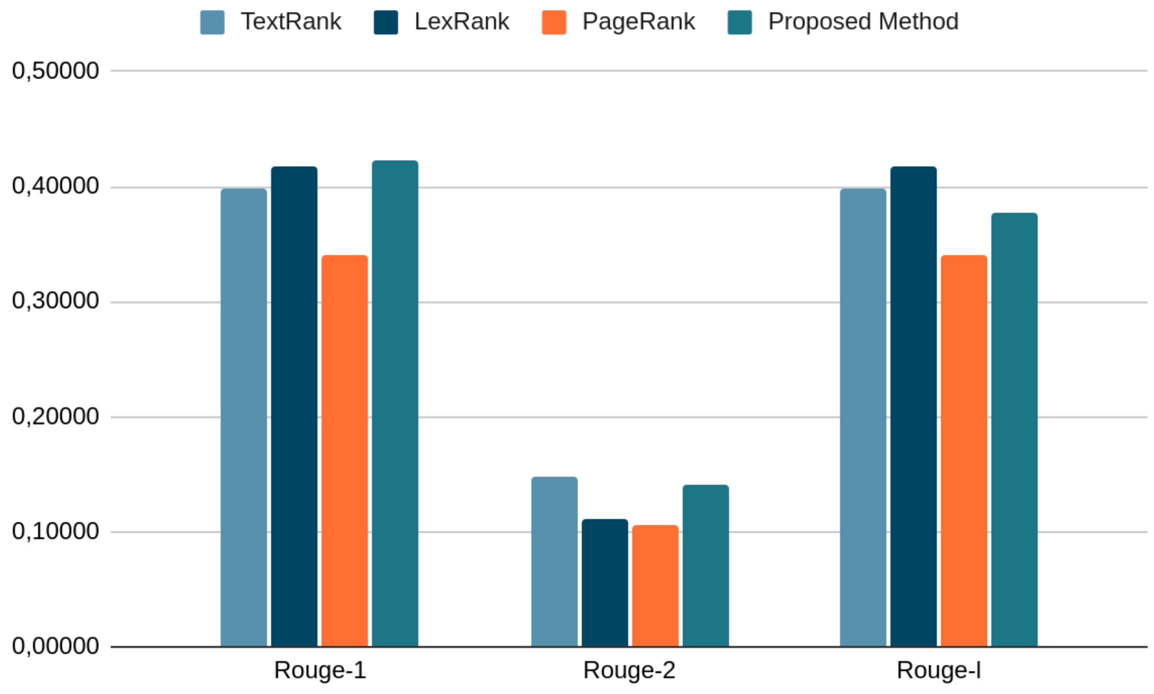

Figure 4.5. Comparison of F-Score Values Based on ROUGE Metricse

5. Conclusion

This proposed novel approach has been realized through the implementation of the graph-based extraction method known as the Malatya Centrality Algorithm. Due to its graph-based nature, it has the potential to generate sentences that are faster and more comprehensible compared to abstractive summaries. The Malatya Centrality Algorithm is renowned for its polynomial approach to solving vertex cover problems in graphs. Therefore, utilizing the Malatya Centrality Algorithm in graph-based methods for text summarization could contribute to achieving more effective results. This new method possesses the ability to infer during the summarization process. The graph-based approach expedites the transformation of the text into a summary by identifying significant nodes within the text. The polynomial nature of the Malatya Centrality Algorithm is optimized for graph analysis, implying that more effective results can be achieved during inference.

This study introduces a new perspective in the field of text summarization. Graph-based extraction methods hold the potential to generate effective and comprehensible summaries even with large datasets. As a result of this study, it is evident that the employed method offers faster and more comprehensible outputs during the text summarization process. The algorithm functions swiftly even with lengthy texts, identifying the most impactful nodes within the text to generate the summary. Furthermore, there is a focus on enhancing sentence-to-sentence semantic coherence by further refining the summarization methods and algorithms. This novel approach could later be hosted on servers, allowing numerous users to access and generate summaries from texts via the internet.

References

Yakut S, Oztemiz F, Karci A(07.12.2022). A New Approach Based on Centrality Value in Solving the Minimum Vertex Cover Problem: Malatya Centrality Algorithm. Computer Science. Volume Vol:7, Issue Issue:2, 81 - 88.

Hark C, Taner Uçkan T, Seyyarer E. , Karcı A(30.09.2019). Metin Özetlemesi için Düğüm Merkezliklerine Dayalı Denetimsiz Bir Yaklaşım dergipark, 8(3).

Hark C, Taner Uçkan T, Karcı A(29.06.2022). A new multi-document summarisation approach using saplings growing-up optimisation algorithms: Simultaneously optimised coverage and diversity

Erhandı, B. (2020). Derin Öğrenme ile Metin Özetleme

Tülek, M. (2007). Türkçe için Metin Özetleme

Kaynar, O., IŞIK, Y. E., GÖRMEZ, Y., & DEMİRKOPARAN, F. (2017). Genetic Algorithmn Based Sentence Extraction For Automatic Text Summarization. dergipark, 3(2).

Khushboo S. Thakkar, R.V. Dharaskar, & M.B. Chandak. (2010). Graph-Based Algorithms for Text Summarization. IEEE. 10.1109/ICETET.2010.104

Güneş Erkan, & Dragomir R. Radev. (2004). LexRank: Graph-based Lexical Centrality as Saliencein Text Summarization.

Ibrahim F. Moawad, & Mostafa Aref. (2013). Semantic graph reduction approach for abstractive Text Summarization. IEEE. 10.1109/ICCES.2012.6408498

Rafael Ferreira, Frederico Freitas, Luciano de Souza Cabral, Rafael Dueire Lins, Rinaldo Lima, Gabriel França, Steven J. Simskez, & Luciano Favaro. (2013). A Four Dimension Graph Model for Automatic Text Summarization. IEEE. 10.1109/WI-IAT.2013.55

Joel Larocca Neto, Alex A. Freitas, & Celso A. A. Kaestner. (2003). Automatic Text Summarization Using a Machine Learning Approach. springer.

Karel Ježek, & Josef Steinberger. (2007). Automatic Text Summarization (The state of the art 2007 and new challenges).

Chirantana Mallick, Ajit Kumar Das, Madhurima Dutta, Asit Kumar Das, & Apurba Sarkar. (2018). Graph-Based Text Summarization Using Modified TextRank. springer.

Rada Mihalcea. (2004). Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization.

Makbule Gulcin Ozsoy, & Ferda Nur Alpaslan. (2011). Text summarization using Latent Semantic Analysis. 10.1177/0165551511408848

Yogesh Sankarasubramaniam, Krishnan Ramanathan, & Subhankar Ghosh. (2014). Text summarization using Wikipedia. sciencedirect.

Rasim ALGULIEV, & Ramiz ALIGULIYEV. (2009). Evolutionary Algorithm for Extractive Text Summarization. Scientific Research.

Naresh Kumar Nagwani, & Dr. Shrish Verma. (2011). A Frequent Term and Semantic Similarity based Single Document Text Summarization Algorithm. researchgate.

Raed Z. Al-Abdallah, & Ahmad T. Al-Taani. (2017). Arabic Single-Document Text Summarization Using Particle Swarm Optimization Algorithm. sciencedirect.

S.A. Babar, & Pallavi D. Patil. (2015). Improving Performance of Text Summarization. sciencedirect.

Arti Jain, Anuja Arora, Jorge Morato, Divakar Yadav, & Kumar Vimal Kumar. (2022). Automatic Text Summarization for Hindi Using Real Coded Genetic Algorithm. mdpi.

Carlos N. Silla Jr., Gisele L. Pappa, Alex A. Freitas, & Celso A. A. Kaestner. (2004). Automatic Text Summarization with Genetic Algorithm-Based Attribute Selection. springer.

Sumya Akter, Aysa Siddika Asa, Md. Palash Uddin, Md. Delowar Hossain, Shikhor Kumer Roy, & Masud Ibn Afjal. (2017). An extractive text summarization technique for Bengali document(s) using K-means clustering algorithm. IEEE. 10.1109/ICIVPR.2017.7890883