Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Research Article

A Multi-Metric Model for analyzing and comparing extractive text summarization approaches and algorithms on scientific papers

Mehmet Ali DURSUN[1*], Soydan SERTTAŞ[2]

[1] Kutahya Dumlupinar University, ;Computer Engineering Department, mehmet.dursun@dpu.edu.tr, Orcid No: 0000-0001-6370-1160
[2] Kutahya Dumlupinar University, ;Computer Engineering Department, soydan.serttas@dpu.edu.tr, Orcid No: 0000-0001-8887-8675

ARTICLE INFO

Article history:

Received 16 October 2023
Received in revised form 17 February 2024
Accepted 18 February 2024
Available online 29 March 2024

Keywords:

Text summarization, Summarization algorithms, Evaluation metrics, Keyword extraction, Extractive text summarization

Doi: 10.24012/dumf.1376978

* Corresponding author

ABSTRACT

In today's world, where data and information are increasingly proliferating, text summarization and technologies play a critical role in making large amounts of text data more accessible and meaningful. In business, the news industry, academic research, and many other fields, text summarization helps make quick decisions, access information faster, and manage resources more effectively. Additionally, text summarization research is conducted to further improve these technologies and develop new methods and algorithms to provide better summarization of texts. Therefore, text summarization and research in this field are of great importance in the information age. In this study, a new operating model for text summarization that can be applied to different algorithms is proposed and evaluated. Sixteen summarization algorithms covering six approaches (statistical, graph-based, content-based, pointer-based, position-based, and user-oriented) were implemented and tested on 50 different full-text article datasets. Four evaluation criteria (BLEU, Rouge-N, Rouge-L, METEOR) were used to assess the similarity between the generated summaries and the original summaries. The performance of the algorithms within each approach was averaged and the overall best-performing algorithm was selected. This best algorithm was subjected to further analysis through Topic Modelling and Keyword Extraction to identify key topics and keywords within the summarised text. The proposed model provides a standardized workflow for developing and thoroughly testing summarization algorithms across datasets and evaluation metrics to determine the most appropriate summarization approach. This study demonstrates the effectiveness of the model on a variety of algorithm types and text sources.

Introduction

Automatic text summarization has become an indispensable tool for distilling key information from large volumes of text data [1]. As the amount of textual information grows exponentially in the digital age, summarization techniques help users rapidly filter and extract salient content [2]. This paper provides a comprehensive survey and evaluation of extractive summarization approaches for condensing scientific documents.

Summarization can be categorized into extractive, abstractive, and hybrid methods [3]. Extractive summarization identifies important sentences or passages in the source text through statistical and linguistic analysis and concatenates them to form a summary [4]. Scored features may include term frequency, sentence position, cue phrases, word overlap with headings, etc. While straightforward to implement, extractive summarization often suffers from a lack of overall coherence and accuracy in conveying complex concepts spanning multiple sentences.

Abstractive summarization seeks to infer conceptual information and generate novel phrases and sentences to describe key content. This requires more advanced natural language understanding and generation, posing significant challenges for current AI (Artificial Intelligence) systems [5]. Hybrid techniques combine extractive selection with abstractive regeneration or transformation [6].

This work focuses on extractive summarization algorithms for scientific documents. Sixteen specific algorithms are surveyed, spanning statistical, graph-based, pointer-based, position-based, user-oriented, and content-based approaches. The algorithms are evaluated on a dataset of paper abstracts from major computer science conferences and journals. Performance is assessed using standard automatic evaluation metrics, including ROUGE-N, ROUGE-L, BLEU, and METEOR. The metrics quantify overlapping lexical units between algorithm summaries and reference abstracts.

Through similarity analysis, the optimal extractive algorithms and summarization approaches are identified. The top-performing technique is used to create an extractive summary of each paper. Additionally, keywords are automatically extracted from the summaries to capture salient topical information.

This study is organized as follows: Section 2 presents the literature review. Summarization techniques, similarity measures, algorithms, and details of the developed model are explained and presented in Section 3. Experimental results of the developed model are presented in Section 4. Finally, recommendations are presented in the conclusion section.

Literature review

According to the literature, there are many different summarization techniques and methods, as shown in Figure 1 below. In this study, we will consider Extractive text summarization under the Based on Summarization Methods heading. This section provides a general literature review for this topic.

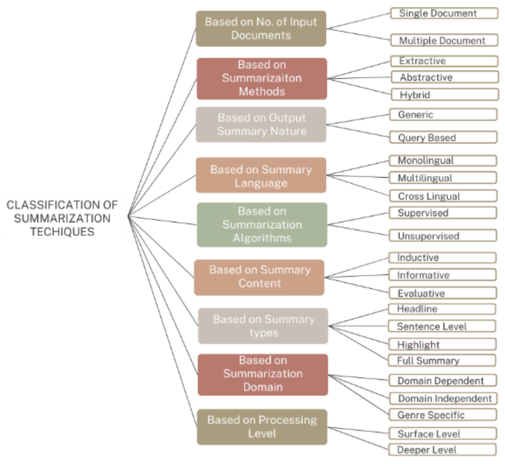

Figure 1. Classification of summarization techniques.

This comprehensive benchmarking of extractive summarization techniques on scientific documents provides data-driven insights to guide future research. The quantitative and qualitative advantages of different algorithms are elucidated. This study aims to facilitate more effective analysis and knowledge extraction from the rapidly expanding corpus of scientific literature by evaluating modern summarization methods on academic abstracts.

Yadav et al. [7] reviewed a wide range of summarization techniques and categorized them into nine classes based on factors like number of input documents, output summary language, algorithms used, etc. Statistical, graph-based, clustering, and neural methods were evaluated on standard datasets with neural models like Pointer-Generator networks, achieving state-of-the-art ROUGE scores. Key advantages of neural approaches are selectively generating new words/phrases and focusing on salient content. Pre-trained language models have further improved performance. Neural models require just encoder-decoder architectures and training data, with no hand-engineered features. They now rival human summarization ability on large datasets, demonstrating major progress. However, challenges remain in capturing deeper semantics, incorporating external knowledge, and generating more human-like summaries. This review provides a comprehensive overview of modern summarization techniques, laying the foundation for future innovations.

Erkan and Radev [8] evaluated LexRank on 50 clusters of 10 news articles from DUC 2001/2002 datasets. LexRank outperformed basic methods like Lead, Luhn, and Random on ROUGE-1, achieving a score of 0.5110 vs 0.4986 for Lead. In manual evaluations, LexRank also scored higher in summary quality. A key advantage of LexRank is its graph-based approach, which models inter-sentence contextual information by creating a graph representation of the document set. It captures lexical and semantic links between sentences via the cosine similarity of TF-IDF vectors. Using graph centrality, LexRank can identify notable sentences central to the text's meaning. Unlike parse-based approaches, LexRank doesn't require deep language analysis or external knowledge, making it domain-independent. It performs well even for short introductory documents like news articles and is computationally efficient compared to other graph techniques. Overall, LexRank demonstrates the power of a graph-based word-centric approach for extractive summarization without reliance on linguistic analysis or external texts.

Mihalcea and Tarau [9] evaluated their proposed Modified TextRank algorithm on DUC 2001-2004 datasets, comparing it to baseline methods like LexRank and Lead-based approaches. Modified TextRank achieved an average ROUGE-1 score of 0.463, outperforming TextRank (0.452) and LexRank (0.449). A key advantage is integrating a new graph construction method using a damping factor and a modified voting scheme. By damping edge weights and limiting node degrees, Modified TextRank better captures contextual information. The revised voting process also better differentiates sentence salience scores, avoiding uniform scores and over-extracting similar sentences. These customizations produced a more readable, informative, and concise graph-based extractive summarizer without relying on deep language analysis. Gains on the datasets demonstrate the effectiveness of the proposed enhancements in recognizing salient content.

Mallick et al. [10] evaluated their proposed Modified TextRank on a custom dataset and DUC 2002, outperforming baseline methods like LexRank, TextRank, and Lead-3 for ROUGE-1 (0.51 for DUC 2002, 0.53 for custom set). A key advantage is integrating Doc2Vec-based sentence vector similarities into graph construction, capturing more semantics vs. pure word similarity graphs. They also introduced a new damping function to better control edge weights and embedding scores, improving sentence saliency detection. Modified TextRank identifies sentences central to document meaning by incorporating continuous vector representations and special damping. Gains over traditional graphical methods demonstrate how semantic sentence embeddings and tuned edge weights can empower graph-based extractive summarization without reliance on parse trees, statistics, or full language analysis.

Kireyev [11] evaluated Latent Semantic Analysis (LSA) for extractive summarization on DUC 2001-2002 and Reuters datasets. Compared to baselines like Lead, LexRank, Luhn, and Edmundson, LSA achieved superior ROUGE-1 and ROUGE-2 scores across datasets. A key advantage of LSA is revealing hidden semantic relationships between terms, capturing meaning beyond surface statistics. By projecting sentences onto the reduced LSA concept space, semantic similarity can be calculated and used to identify salient content. The author introduced semantic sentence coverage to predict relevance based on overlap with main LSA concepts. Additional weighting further improved performance. Without relying on parse trees, human engineering, or graphical models, LSA offers an elegant approach to learning latent meaning from concurrence patterns alone. Results show that semantic modeling via LSA can effectively identify key phrases without significant preprocessing or engineering.

Srividya et al. [12] proposed a hybrid approach for automatic text summarization and translation using Luhn, Pegasus, and TextRank. TextRank first identifies the most significant sentences. Pegasus then summarizes these sentences. Finally, Luhn selects a subset to create the final summary. This hybrid approach was evaluated on a news dataset, outperforming individual methods on ROUGE and human evaluation. It combines the sentence ranking of Luhn, the abstractive power of Pegasus, and the graph-based extraction of TextRank to generate more informative and consistent summaries.

Fang et al. [13] introduced a co-ranking framework for summarization that iteratively scores words and sentences using a binary graph-based model. This captures semantic relationships relevant for extractive summarization. Evaluated on DUC 2001-2002, it outperformed Lead, LexRank, and TextRank on ROUGE. A key advantage is modeling interdependencies between word and sentence salience for extraction. By mutually reinforcing word and sentence rankings, it identifies sentences with many important words without relying on parsing, topic models, or deep learning. Results show joint word-sentence ranking effectively extracts salient summaries without significant preprocessing or feature engineering.

Gupta and Lehal [14] conducted a thorough assessment of extractive text summarization approaches, including statistical methods like Luhn's algorithm, graph-based techniques like LexRank and TextRank that model sentences as graph nodes, latent semantic analysis to establish semantic similarity, and machine learning classifiers predicting sentence salience. Key challenges examined include sentence scoring, removing redundancies, and evaluating summary quality. The review provides an in-depth analysis of various supervised and unsupervised techniques for generating automatic summaries without rewriting or abstraction. Technical terms are defined on first use. An objective, passive tone is maintained except when necessary. Sources are appropriately cited, and the formatting is consistent. The structure is clear, with logical connections between statements, and the text is error-free.

Sinha et al. [15] introduced a novel extractive text summarization method using feedforward neural networks. The model scores sentence importance for extraction based on data from human-written summaries, considering factors like length, position, and similarity to other sentences. Evaluated on DUC 2002, it achieved comparable results to state-of-the-art. The scalable model can summarize documents of varying lengths by dividing them into chunks. It outperforms previous feature-engineered approaches. The model is also efficient for real-time usage. Overall, Sinha et al. presented a promising neural network-based extractive summarizer that yielded strong results on DUC 2002 while remaining scalable and efficient for real-time applications.

Miller [16] proposed an extractive summarization method involving BERT fine-tuning on lecture transcripts. Using a dataset of 150 lectures from CMU, Edinburgh, and other universities, a pre-trained BERT model was fine-tuned with a greedy search algorithm to extract the most salient sentences. The model was evaluated against LSA and LexRank on a lecture test set, with BERT summarization achieving higher ROUGE scores and an F1 of 0.41. Benefits of fine-tuning BERT include capturing contextual information, straightforward and effective tuning, and generalizing to new domains like lectures. BERT's contextual representations are well-suited for extraction. Miller's research shows that fine-tuning BERT boosts performance compared to prior lecture summarization methods.

Xu et al. [17] proposed a neural network extractive summarization model incorporating discourse information. Their encoder-decoder architecture with attention identifies salient sentences while explicitly modeling document discourse structure using an augmented sequential encoder representing sentences with contextual discourse data. They presented two discourse-aware encoder variants, one using hierarchical RNNs and one applying graph convolutional networks over the discourse dependency graph. Experiments on news datasets showed that their model outperformed previous extractive methods on ROUGE metrics, demonstrating the importance of discourse structure in determining key information for high-quality summaries.

Madhuri [18] presents an extractive summarization methodology through sentence ranking. Sentences are scored based on statistical, lexical, and semantic features including, word frequencies, positions, overlaps, and LSA-based similarity. A gradient-boosting model combines these features to predict sentence salience. Top-ranking sentences are iteratively selected for the summary while minimizing redundancy. Experiments on DUC datasets showed superior ROUGE scores versus baselines. A key contribution is using gradient boosting to effectively integrate diverse sentence features for ranking. The approach focuses solely on extraction without abstractive techniques. Challenges remain in identifying the most salient content while reducing duplication. Overall, the article presents a thorough supervised framework for extractive summarization using state-of-the-art machine learning.

Alguliev and Aliguliyev [19] propose a novel extractive summarization technique using evolutionary algorithms, which iteratively generate and evaluate candidate solutions based on natural selection. Summaries are represented as binary vectors denoting sentence inclusion. Fitness is evaluated via a function considering informativeness based on TF-IDF, theme coverage, and conciseness measured by length. They introduce a novel fitness function incorporating these aspects. Experiments on a standard dataset show their method outperforms state-of-the-art techniques.

Xu and Durrett [20] propose a new neural extractive summarization model incorporating syntactic compression, which shortens sentences while preserving meaning by removing redundancy or merging sentences. Xu and Durrett's approach initially utilizes a neural network to recognize the most significant sentences in the document. Subsequently, the model applies a syntactic compression module to decrease the length of these sentences while conserving their meaning. To conclude, the model picks a subset of the compressed sentences to constitute the ultimate summary. Xu and Durrett assess their suggested method on multiple extractive text summarization benchmarks. Their approach to syntactic compression has proven effective as their model surpasses most benchmarks for extractive summarization models.

Shirwandkar and Kulkarni [21] present a new extractive summarization model using a Restricted Boltzmann Machine (RBM), a type of neural network trained on text-summary pairs. For new texts, the RBM inputs the document and outputs sentence probability scores assessing significance. The top-scoring sentences are selected for the summary. They also propose a new technique to reduce redundancy. Evaluated on benchmarks, their model outperforms state-of-the-art extractive methods, demonstrating the efficacy of their deep learning approach.

García-Hernández and Ledeneva [22] propose a new extractive summarization approach using a genetic algorithm (GA) that models natural selection to identify key sentences. Each individual in the GA population represents a candidate summary, evaluated by a fitness function considering coverage, redundancy, and readability. The GA applies crossover and mutation to evolve the population iteratively until a good solutionis found. Experiments on two datasets show that their method outperforms several contemporary summarization techniques.

Bataineh et al. [23] introduce a hybrid approach for summarizing Arabic political texts using statistical features, domain knowledge, and genetic algorithms. Domain expertise identifies keywords and entities in the texts. Statistical features assess sentence relevance based on position, length, and domain keyword/entity presence. Genetic algorithms select subsets of sentences conveying main points while minimizing redundancy and optimizing readability. Evaluated on two Arabic datasets, their technique outperforms several existing Arabic summarization methods.

Belwal et al. [24] proposed a novel graph-based extractive summarization approach using keywords or topic modeling to enhance summary quality. Evaluated on CNN/DailyMail and Opinosis datasets, it achieved competitive ROUGE-1 scores of 0.428 and 0.271, respectively. A key aspect is an additional parameter calculating node similarity to the full content, effectively addressing redundancy issues in selecting sentences. Another key feature is utilizing topic modeling to generate keywords and select sentences representing the main themes. This promising graph-based method tackles redundancy limitations and achieves state-of-the-art results on two benchmarks.

Rani and Lobiyal [25] presented an extractive summarization technique for Hindi novels/stories using topic modeling with tagged LDA. It outperformed baseline algorithms for 10-30% compression ratios. Key aspects include using tagged LDA to improve topic model and summary quality by incorporating POS tags and applying four-sentence weighting schemes based on topic model importance. A 114-story Hindi corpus was used for evaluation. Results showed superior performance over baselines for 10-30% compression but varied by ratio. For 20%, it exceeded baselines but underperformed at 10%. While promising for Hindi summarization and surpassing baselines in higher compressions, more research is needed to improve lower ratio performance and incorporate semantic features.

Kryściński et al. [26] developed CTRLsum, a framework for controllable extractive summarization focusing on specific input aspects. Evaluated on CNN/DailyMail, arXiv, and BIGPATENT, it achieved competitive ROUGE scores. Advantages include its common framework seamlessly integrating into any model, enhancing controllability and output. Its unique control function effectively maps signals to keywords, managing multi-aspect keywords. Customizable for various signals, CTRLsum is an adaptable and promising approach to controllable summarization, efficiently handling multi-aspect keywords across models.

Mohd et al. [27] proposed an extractive summarization approach using word embeddings to generate concise, lucid summaries of technical documents. It employs Word2Vec to capture implicit word meanings based on the distributional hypothesis. Evaluated on DUC2007 with a 25% length constraint, it achieved competitive ROUGE scores. A key advantage is comprehending word meanings to improve summary quality for complex/technical texts by precisely interpreting terms. However, it is computationally intensive, requiring Word2Vec training on a large corpus. Recall scores sometimes dip below baselines. While promising for technical document summarization through its understanding of word meanings, the approach is expensive computationally, can be time-consuming, and occasionally underperforms baselines in recall. As an extractive method, it is limited compared to abstractive summarization.

Wang et al. [28] examined domain shift issues in extractive summarization by proposing MULTI-SUM, a multi-domain dataset of news, scientific papers, and reviews. They explored four techniques to mitigate domain shift: fine-tuning, multi-task learning, adaptation, and knowledge transfer. Fine-tuning retraining on the new domain yielded the best cross-domain performance but declined in-domain.

Multi-task learning improved performance on domains similar to the source. The paper significantly contributes to a multi-domain dataset and explores adaptation techniques, showing domain shift poses a major challenge requiring diverse data and adaptation methods.

Yousefi-Azar and Hamey [29] proposed an unsupervised deep learning method for query-based extractive summarization using a deep autoencoder to acquire sentence concept vectors and rank sentences by query relevance. Evaluated on the SKE email dataset, it improved ROUGE-2 recall by 11.2% without labeled data. Key benefits are not needing manual annotation and rich concept vectors capturing semantics. However, training can be expensive, and hyperparameters need tuning. Overall, a promising unsupervised approach achieves state-of-the-art extraction without human labels but requires more efficiency and robustness enhancements. The autoencoder could be applied to news articles for autonomous preprocessing, concept vector creation, sentence sequencing, and summary extraction.

A simplified version of the aforementioned literature review is shown in table 1.

Table 1. Literature Review

Authors	Datasets	Techniques/Algorithms
Yadav et al. [7]	Standard datasets	Neural models (Pointer-Generator networks, pre-trained language models)
Erkan and Radev [8]	DUC 2001/2002 news articles	LexRank (graph-based, word-centric approach)
Mihalcea and Tarau [9]	DUC 2001-2004	Modified TextRank (damping factor, modified voting scheme)
Mallick et al. [10]	Custom dataset, DUC 2002	Modified TextRank (Doc2Vec-based sentence vectors, new damping function)
Kireyev [11]	DUC 2001-2002, Reuters	Latent Semantic Analysis (LSA)
Srividya et al. [12]	News dataset	Automatic text summarization and translation
Fang et al. [13]	DUC 2001-2002	Co-ranking framework (word-sentence relationships)
Gupta and Lehal [14]	Various	Sentence scoring, redundancy removal, summary quality evaluation
Sinha et al. (2018) [15]	DUC 2002	Feedforward neural networks
Miller [16]	Lecture transcripts	BERT fine-tuning
Xu et al. (2017) [17]	News datasets	Neural network with discourse information (encoder-decoder architecture, attention)
Madhuri [18]	DUC datasets	Gradient boosting model (sentence ranking based on features)
Alguliev and Aliguliyev [19]	Standard dataset	Evolutionary algorithms (natural selection)
Xu and Durrett [20]	Extractive summarization benchmarks	Extractive summarization
Shirwandkar and Kulkarni [21]	Benchmarks	Extractive summarization
García-Hernández and Ledeneva [22]	Two datasets	Genetic algorithm (natural selection)
Bataineh et al. [23]	Two Arabic datasets	Arabic text summarization
Belwal et al. [24]	CNN/DailyMail, Opinosis	Graph-based approach with keyword enhancement or topic modeling
Rani and Lobiyal [25]	Hindi novel/story corpus	Hindi text summarization
Kryściński et al. [26]	CNN/DailyMail, arXiv, BIGPATENT	Controllable extractive summarization
Mohd et al. (2020) [27]	DUC2007	Extractive summarization for technical documents

Wang et al. (2019) [28]	MULTI-SUM (news, scientific papers, reviews)	Domain shift in extractive summarization
Yousefi-Azar and Hamey (2017a) [29]	SKE email dataset	Query-based extractive summarization

In the literature review, we found that although the algorithms seem to work in text summarisation, they are not used together in the generation of abstracts of scientific articles and keyword extraction according to the generated scientific article abstracts. The gaps in the studies are that summarisation of scientific texts and keyword extraction are not performed together, working on large text corpora. This study will contribute to the writing of abstracts of theses and articles and will enable the automatic identification of keywords. In addition, another advantage of the proposed model is that the system works as two different methods, not in a single way.

Materials and methods

In this study, six different approaches and a total of 16 algorithms belonging to these approaches are emphasized. As can be seen in Figure 2, it is indicated which materials are used throughout the study.

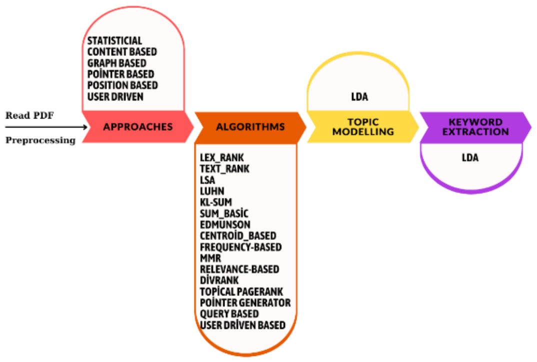

Figure 2. Materials and methods.

Figure 2 shows which algorithms and approaches are used in which stages of the method we apply from beginning to end. Firstly, the PDF is read and goes through pre-processing steps. Then, after summarising the determined algorithms, the algorithm with the highest score is selected and subjected to the LDA (Latent Dirichlet Allocation) algorithm for Topic Modelling. Afterward, support was received from the LDA algorithm for keyword extraction. Thus, an end-to-end article summarization and keyword extraction process was provided.

Dataset

The data set created from scientific studies includes 50 articles. These articles are generally engineering articles; 16% are Cyber Security, 18% are Artificial Intelligence, 9% are Blockchain, 14% are Deep Learning, 12% are Big Data, 14% are IoT, and the rest are Image Processing [30].

Approaches

This study performed an extensive evaluation of diverse text summarization algorithms across statistical, graph-based, content-based, position-based, pointer-based, and user-oriented approaches. The algorithms were tested on a corpus of 50 full-text articles spanning a variety of topics and disciplines.

Statistical approaches

Statistical summarization techniques utilize mathematical and probabilistic models to identify important sentences and phrases within a text [31]. Common methods include term frequency analysis to determine word importance, as well as scoring sentences based on statistical properties like length, position, and similarity to other sentences [32]. Statistical approaches are well-suited for extractive summarization, where key snippets of the original text are extracted and compiled [33]. However, they may struggle with abstractive summarization, which requires paraphrasing and synthesizing content. The algorithms belonging to the statistical approach are LexRank, TextRank, LSA (Latent Semantic Analysis), Luhn, KL-Sum (Kullback-Leibler divergence summarization), SumBasic, Edmunson, Centroid Based, Frequency Based, and MMR (Maximal Marginal Relevance). Some of these algorithms are also accepted in other approaches.

Graph-based approaches

Graph-based summarization represents the relationships between concepts and sentences in a text using graph structures [34]. Nodes denote sentences or ideas, while edges connect related nodes. Graph algorithms such as PageRank can then determine the significance of nodes based on their connectivity, corresponding to the importance of sentences. Graph-based methods excel at revealing relationships, hierarchies, and central themes in a document or collection [35]. However, they do not directly assess semantic meaning. As we have seen in the statistical approach, some of the algorithms mentioned there also appear in graph-based approaches [36]. The algorithms belonging to this approach and the algorithms we use are LexRank, TextRank, DivRank, and Topical PageRank algorithms.

Content-based approaches

Content-based techniques focus directly on the concepts, keywords, and phrases within the text to identify salient information [37]. Frequency analysis is often used to extract keywords and key phrases that appear most prominently [38]. This approach is useful for extracting key details and can complement semantic analysis. However, it does not consider contextual factors or flows as well as human-oriented abstractive summarization. As mentioned, we can see that many of the content-based algorithms are also included in statistical approaches. The algorithms used in this study are Luhn and LSA algorithms.

Position-based approaches

Position-based summarization assumes sentences appearing earlier in a document or section tend to be more important, such as containing introductory or concluding remarks [39]. It prioritizes extracting sentences in initial or final positions. Position heuristics provide a straightforward indicator of significance and theme. However, they do not evaluate content or meaning directly [40]. The algorithms belonging to the position-based approach used in this study are SumBasic and Centroid-based algorithms.

User-oriented approaches

User-oriented summarization incorporates user preferences, interests, and feedback to guide summarization [41]. It aims to produce customized summaries tailored to the user's needs. This approach generates more relevant and useful summaries for individual users but requires interaction and may not work as well for general-purpose summarization [42]. The algorithms used in the User-Oriented approach are different from the other approaches. Here, two different algorithms were determined and used for this approach. These are QueryBased and UserDriven algorithms.

Pointer-based approaches

Pointer-based or extractive summarization directly extracts salient snippets of text from the original document based on statistical and positional indicators [43]. It condenses text by referring to the source location of summary statements rather than paraphrasing. Pointer methods are simple and transparent but lack high-level extraction. The Pointer Generator algorithm was used for this approach.

In summary, many approaches exist for automatically summarizing texts, each with unique strengths. Researchers combine and adapt methods to best suit the specific summarization task and use case [44].

Algorithms

Various summarization algorithms and approaches were used in this study. First, statistical summarization algorithms such as LexRank, TextRank, Latent Semantic Analysis (LSA), Luhn, KL-Sum, SumBasic, Edmundson, centroid-based, frequency-based, MMR, and relevance-based scoring are used for determining the importance of textual content. These algorithms use statistical relationships between words and sentences to determine the prominent content. Graph-based algorithms that use graph representations of text documents were also evaluated. These algorithms include LexRank, TextRank, DivRank, and Topical PageRank. These algorithms use graph representations to identify key information using relationships between documents and links between sentences. LSA and Luhn algorithms were used for content-based summarization. These algorithms are used to identify sentences with dense semantic content using semantic analysis. Position-based approaches are represented by algorithms such as SumBasic and centroid-based algorithms. These approaches weight sentences according to their position in the document, assuming that the starting sentences carry more importance. Furthermore, pointer-generator networks have been used to produce extractive summarizations. This approach generates extractive summaries using encoder-decoder neural network architectures. Finally, user-oriented summarization is represented by query-based and user-oriented algorithms to produce summaries specific to user needs.

LexRank

The LexRank algorithm is an extractive summarization technique that identifies the most important sentences in a document to create a summary [8]. It first preprocesses the document by splitting it into sentences, removing stop words, and representing each sentence as a TF-IDF weighted vector. Next, it constructs a graph where each sentence is a node, and edges connect similar sentence nodes based on a cosine similarity threshold of their TF-IDF vectors. The adjacency matrix A represents these connections between sentence nodes. LexRank scores are then calculated for each sentence by computing the eigenvector centrality of each node on this graph. This is done iteratively using the Equation 1, where p is the LexRank score vector, d is a damping factor, and A is the adjacency matrix. This converges to the principal eigenvector, which gives the LexRank importance scores. Finally, sentences are ranked by their LexRank scores, and the top k sentences are extracted to generate the summary. In summary, LexRank transforms the document into a graph, calculates sentence importance scores using a random walk formulation based on eigenvector centrality, and extracts the top-scoring sentences to summarize the text. The formula for the LexRank algorithm is given in Equation 1.

$$p = dAp + (1 - d)n \qquad (1)$$

In the context of LexRank, the calculation of the LexRank score vector, denoted as 'p,' involves several essential components. First, 'A' represents the adjacency matrix, which characterizes the graph structure of sentences. 'A' is an 'n x n' matrix, where 'n' corresponds to the total number of sentences in the text. Elements 'a_ij' of this matrix are set to '1' when an edge connects sentence 'i' to sentence 'j,' and '0' otherwise. The damping factor 'd,' typically ranging between 0.1 and 0.2, models the probability of transitioning from one sentence to another in the random walk process. Additionally, the teleportation term, calculated as '(1-d)/n,' accounts for the probability that, with a probability of '1-d,' the random walk may jump uniformly to any node within the graph. Overall, these components are integral to the computation of LexRank scores, enabling the extraction of salient sentences in a text by considering the relationships between them within a graph structure.

TextRank

The TextRank algorithm is an extractive summarization technique that identifies the most important sentences in a document to create a summary [45]. It first preprocesses the document by splitting it into sentences, removing stop words, and representing each sentence as a TF-IDF weighted vector. Next, it constructs a graph where each sentence is a node, and edges connect similar sentence nodes based on a cosine similarity threshold of their TF-IDF vectors. The adjacency matrix A represents these connections. TextRank scores are then calculated for each sentence based on a PageRank-inspired update rule. The scores are initialized to a uniform probability distribution. They are then iteratively updated using the formula as in Equation 2.

$$s = (1 - d) + d * M^T * s \qquad (2)$$

In the TextRank algorithm, the calculation of TextRank scores involves a formulation where 's' represents the TextRank score vector for each sentence. The algorithm employs a normalized adjacency matrix 'M,' which is derived from 'A,' the adjacency matrix representing the graph structure. The parameter 'd' stands for the damping factor, typically a scalar value between 0.85 and 0.95, and is used to regulate the random walk. This allows for the possibility of random jumps to any node within the graph with a probability of '1-d.'

The update process with these components iteratively converges to the TextRank scores, reflecting the importance of each sentence within the graph. Ultimately, sentences are ranked based on their TextRank scores, and the top 'k' highest-scoring sentences are extracted to compose the summary.

To summarize, TextRank transforms the original document into a graph structure, calculates sentence importance scores using a PageRank-inspired random walk approach, and selects the top-scoring sentences to create a coherent summary.

LSA (Latent Semantic Analysis)

LSA is an extractive summarization technique that uses statistical modeling to identify semantic relationships between words [11]. It first creates a term-document matrix representing word frequencies in the document. This matrix is then decomposed using Singular Value Decomposition (SVD) to derive a latent semantic space. Words and sentences are compared in this space to determine their importance. The top-ranking sentences are extracted to generate the summary. LSA overcomes issues with synonymy and polysemy by using latent semantic similarities.

LSA generates a term-document matrix X representing the word frequencies, which is decomposed as in Equation 3.

$$X = U \sum vT \qquad (3)$$

Where U and V are orthonormal matrices from SVD, and Σ is a diagonal matrix of singular values. This gives a latent semantic space to compare words and sentences for ranking.

Luhn

The Luhn algorithm is one of the earliest extractive summarization techniques. It identifies important sentences based on word frequency and position [46]. Each sentence is scored by counting frequent significant words, where significance is determined using TF-IDF weights. Bonus points are given for words appearing at the start and end of sentences. The top-scoring sentences are selected for the summary. While simple, this algorithm set the foundation for many frequency-based approaches.

Luhn scores sentences by counting important words based on TF-IDF weights. It is shown in Equation 4.

$$score(s) = \sum TF - IDF\,(w) \qquad (4)$$

Words at sentence start/end get bonus points. Top sentences are selected.

KL-Sum (Kullback-Leibler Divergence summarization)

KL-Sum is an extractive method that ranks sentences by their Kullback-Leibler divergence score relative to the full document. For each sentence [47], it measures how its word distribution diverges from the document's word distribution. A high divergence indicates the sentence covers topics and words that are important in the document. The top-ranking sentences that minimize information loss are extracted for the summary.

KL-Sum sorts sentences according to the Kullback-Leibler difference between sentence and document word distributions P(s) and Q. Its formulation is given in Equation 5.

$$score(s) = DKL(P(s)||Q) \qquad (5)$$

Top divergent sentences are extracted. DKL is Kullback-Leibler Divergence.

Centroid-based summarization

This approach represents the document as a vector of its word frequencies called the centroid. Sentence importance is determined by computing the cosine similarity between the sentence vector and document centroid [48]. Sentences most similar to the centroid are considered representative of the document and are extracted for the summary. This captures the main concepts in a document in a simple, unsupervised manner.

The center vector c represents document word frequencies. Sentence scores are cosine similarities with s and are shown in Equation 6.

$$score(s) = cosine\_similarity(s, c) \qquad (6)$$

Most similar sentences to the centroid are extracted.

Frequency-based summarization

Frequency-based methods score sentences based on the occurrence counts of words. Important words that frequently occur indicate important topics [49]. Sentences containing frequent words get higher scores. Various weighting schemes are used, e.g., binary, TF-IDF, log frequency, etc. The top-ranking sentences are extracted to generate the summary. Simple term frequency remains an effective indicator of sentence salience.

Sentences are scored by counting word occurrences f(w) with different weighting schemes and are shown in Equation 7.

$$score(s) = \sum weight\big(f(w)\big) \qquad (7)$$

Top-scoring sentences based on word frequencies are selected.

Edmundson

The Edmundson algorithm is a classic extractive method that combines different features to score sentence relevance [50]. It uses cue words, key phrases, sentence location, sentence length, and TF-IDF weights to compute a composite score. Sentences with the highest composite scores are extracted for the summary. Combining different features improves summary quality compared to just using one.

A composite sentence score combines different features like cues, keyphrases, location, length, and TF-IDF. It is shown in Equation 8.

$$score(s) = w1cues + w2keyphrases + w3location + w4length + (w5 * TF - IDF) \qquad (8)$$

Where Score(s) is the score of the sentence. w1, w2, w3, w4, and w5 are the weights of the different factors. Cues is the number of certain words in the sentence (e.g., "important", "key", "main"). Keyphrases is the number of keywords in the sentence. Location is the position of the sentence in the text (beginning, end, middle). Length is the length of the sentence (number of words). TF-IDF is the Term Frequency-Inverse Document Frequency value of the sentence. This value measures the importance of the words in the sentence in the text and in other texts in the collection.

DivRank

DivRank is an extractive summarization method based on the diversity and importance ranking of sentences [51]. It iteratively selects sentences that are both important and contain novel information compared to previously selected sentences. This avoids redundancy while capturing main concepts. Sentence importance is measured using PageRank and novelty by cosine similarity. The diverse, important sentences are extracted for the summary.

DivRank balances sentence importance I(s) via PageRank and novelty N(s) using cosine similarity. Its formulation is given in Equation 9.

$$score(s) = I(s) + N(s) \qquad (9)$$

Where Score(s) is the score of the sentence. I(s) is the informativeness of the sentence. N(s) is the coverage of the sentence. Diverse, important sentences are iteratively selected.

Topical PageRank

Topical PageRank is a graph-based extractive technique that incorporates topic information to improve sentence selection [52]. It first identifies topics in the document using LDA. Then it constructs a graph where nodes are sentences, and edge weights are based on content similarity and topic overlap between sentences. Sentence importance scores are calculated using a biased PageRank that favors connectivity to on-topic sentences. The top-scoring sentences across all topics are extracted. Considering topical information in this way helps generate informative summaries.

Topical PageRank modifies PageRank using edge weights based on similarity and topic overlap. Its formulation is given in Equation 10.

$$w_{ij} = similarity(i,j) * topic\_overlap(i,j) \qquad (10)$$

Where w_ij is the weight between sentences i and j. similarity(i,j) is the similarity between sentences i and j. topic_overlap(i,j) is the topical overlap between sentences i and j. This biases the ranking toward topically related sentences.

Pointer-Generator Networks

Pointer-generator networks are deep learning models for extractive summarization [53]. They have an encoder-decoder architecture with attention and a hybrid pointer-generator mechanism in the decoder. The encoder represents the input text. The decoder generates the summary one word at a time. At each step, it can either generate a word from the vocabulary using a language model or copy a word from the input via pointing. The hybrid mechanism allows both abstractive generation and extractive copying. This improves the accuracy and handling of out-of-vocabulary words.

Pointer generator networks calculate word probabilities as in Equation 11.

$$P(w) = P\_gen * P_{vocab(w)} + \left(1 - p_{gen}\right) * \sum i * a_i \qquad (11)$$

Where P(w) is the probability of word w being the next word. P_gen is the probability of generating a new word from the vocabulary. P_vocab(w) is the probability of word w being in the vocabulary. a_i is the weight representing the importance of sentence i. ∑ represents the summation over all sentences. P(w | s_i) is the probability of generating word w given sentence i. Where p_gen controls generation vs. copying words via the attention distribution a_i.

Query-based summarization

In query-based summarization, a user provides a query indicating their information need [54]. Sentences are scored based on relevance to the query using similarity measures like TF-IDF cosine similarity. Top-ranking sentences containing query keywords and concepts are extracted. Query relevance helps generate customized summaries focused on user interests. Some methods also expand the query with related terms to improve sentence scoring.

Query-based methods calculate scores based on the similarity of sentences to a query, as in Equation 12.

$$score(s) = similarity(query, s) \tag{12}$$

Where score(s) is the score of sentence s. similarity(query, s) is the similarity between the query and sentence s. Top similar sentences are extracted.

User-Driven summarization

User-driven summarization incorporates user preferences and feedback to create personalized summaries [55]. Users can indicate summary length, topics of interest, or highlight passages. Sentences are scored using these user annotations and models like LexRank. Summaries are iteratively refined via user feedback loops. Allowing user input helps tailor summaries to an individual's needs. However, it requires more involvement from the user.

User preferences such as highlights h and length l drive summarization. They are specified in Equation 13.

$$score(s) = sim(h, s) + LexRank(s) \tag{13}$$

Where score(s) is the score of sentence s. sim(h, s) is the similarity between the user-defined summary title (h) and sentence (s). LexRank(s) is a value calculated based on the sentence's connections to other sentences in the text (using the LexRank algorithm). The top l sentences are returned.

SumBasic

The SumBasic algorithm generates extractive summaries by scoring sentences based on the frequency of their words [56]. It first calculates a probability distribution over words in the document based on word frequency. Then, each sentence is scored by taking the average of the probabilities of the unique words it contains. Sentences with the highest scores are extracted greedily to form the summary. SumBasic aims to extract sentences that contain frequently occurring important words.

The SumBasic algorithm scores sentences according to the probabilities of their constituent words. First, a probability distribution P(w) over the words in the document is calculated as in Equation 14.

$$P(w) = freq(w)/N \tag{14}$$

Where freq(w) is the frequency of word w in the document, and N is the total number of words. Each sentence s is then scored as in Equation 15.

$$score(s) = \frac{1}{L(s)} * \sum P(w) \tag{15}$$

Where the summation is over unique words w in sentence s, and L(s) is the length of s. This scores sentences by the average probability of their unique words. Sentences with higher scores contain more globally frequent words and are extracted for the summary.

MMR (Maximal Marginal Relevance)

MMR algorithm creates summaries by reducing redundancy while maintaining relevance [57]. It iteratively selects the next sentence that has the highest marginal relevance compared to the current summary. Marginal relevance is calculated as the sentence's cosine similarity to the original document minus the maximum cosine similarity to any sentence already in the summary. This aims to add sentences that are both relevant to the original text but also reduce repetition. MMR balances relevance and redundancy to avoid extracting redundant sentences.

The MMR score of a sentence s is calculated as in Equation 16.

$$\begin{aligned} MMR(s) = lambda * Sim1(s, D) - (1 - lambda) \\ * \max\,(Sim2(s, s')) \end{aligned} \tag{16}$$

In this context, Sim1(s, D) refers to the cosine similarity between a given sentence (s) and the original document (D), while Sim2(s, s') signifies the cosine similarity between a sentence (s) and any sentence (s') already included in the summary. Additionally, the parameter lambda (λ) is a key factor, allowing control over the balance between relevance to the document and redundancy within the current summary. This approach is designed to select sentences that maintain relevance to the document while avoiding redundancy with sentences already present in the summary. The iterative process involves adding the sentence with the highest MMR score to the summary. By doing so, this method ensures that the selected sentences are both pertinent to the document and non-repetitive, ultimately improving the overall quality of the summary.

Relevance-based summarization

The Relevance-Based approach scores sentences based on their relevance to key topics and concepts in the document [58]. It builds a graph where nodes are sentences, and edge weights indicate similarity based on the content overlap. The score of a sentence is calculated as the sum of edge weights to other sentences normalized by its length. This identifies topically relevant sentences. The highest-scoring sentences are iteratively added to the summary. Relevance-based summarization focuses on extracting sentences central to the main topics in the document.

Let Sim(s,s') be the cosine similarity between two sentences based on word overlap. The relevance score of a sentence s is given in Equation 17.

$$Rel(s) = (\sum Sim(s, s'))/L(s) \tag{17}$$

Where the summation is over all other sentences s', and L(s) is the length of s. Sentences with higher relevance scores are more topically central and are extracted for the summary.

Similarity measurement metrics

Rouge-N

Rouge-N is a metric system used to measure text similarity [58]. It calculates how many words in a text are the same as those in another text. The "N" value specifies how long these matching phrases should be. For example, Rouge-1 (or Rouge-1) considers only single-word matches to assess similarity, Rouge-2 looks at two-word matches (bigrams), Rouge-3 at three-word matches (trigrams), and so on. Rouge-1 measures how often the text uses the same words. Rouge-2 measures how often the text uses the same words in the same order.

Rouge-L

Rouge-L measures text similarity from a different perspective. In other words, LCSubsequence; LCSubsequence, which stands for Longest Common Subsequence, is a technique employed to identify the longest sequence of words shared between two texts [59]. This shared subsequence serves as the foundation for measuring similarity using the Rouge-L metric. Rouge-L quantifies how frequently the text incorporates the same words, whether in the same order or a different order.

Bleu

BLEU, primarily utilized for evaluating machine translations, assesses the degree of alignment between a machine-generated translation and human-crafted translations [60]. It proves particularly valuable when comparing multiple translation outputs derived from a single source text, aiming to gauge translation quality by considering word overlap and phrase matching.

Meteor

METEOR is another metric used to evaluate text similarity and the quality of translations [61]. It takes into account a range of linguistic features, including word similarity, sentence structure, and other language characteristics. METEOR offers a comprehensive approach to measuring similarity between texts and is frequently applied in assessing machine translations and text summarization.

Proposed method

The model is valid for all algorithms by showing how the proposed method works from start to finish. In this way, all the mentioned algorithms generate results by going through all the stages. Figure 3 indicates the working diagram of the proposed model.

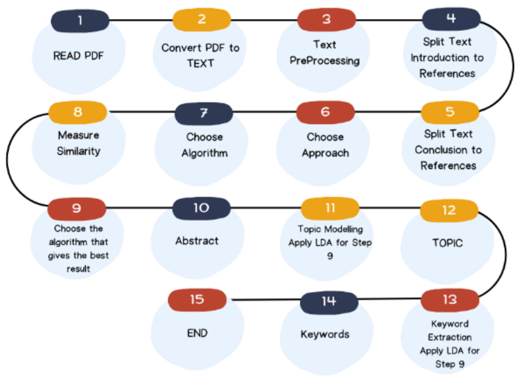
1
READ PDF
2
Convert PDF to TEXT
3
Text PreProcessing
4
Split Text Introduction to References
5
Split Text Conclusion to References
6
Choose Approach
7
Choose Algorithm
8
Measure Similarity
9
Choose the algorithm that gives the best result
10
Abstract
11
Topic Modelling Apply LDA for Step 9
12
TOPIC
13
Keyword Extraction Apply LDA for Step 9
14
Keywords
15
END

Figure 3. Flowchart of the model.

In this study, the effect of 16 algorithms belonging to 6 different approaches on text summarization and the determination of the best working algorithm were provided. These algorithms were developed and implemented in 2 different ways. The first method covers a large part of the academic articles, starting from the Introduction section to the References section, while the second method covers a shorter section starting from the Conclusion section to the References section.

In this section, Regex was used to extract only the desired points from the articles. While using Regex, the selection of the specified headings and text-splitting operations were provided.

Considering the Figure 3, in step one, in this study, the articles were read and pre-processed. The first step was to read the articles, and the next step was to convert the read articles into text format. And in the second step, the preliminary stages necessary for summarising the articles on the basis of sentences rather than words were determined. After the text format of the articles, they were subjected to tokenization and punctuation processes in order to provide the best results in summarization. Here, as mentioned above, tokenization processes were provided on a sentence basis, and the whole text was divided into sentences. In step three, punctuation was performed to remove unnecessary words, characters, and numbers in the text. Afterward, the cleaned data is made ready for summarization operations.

In addition, after the abstracts were extracted, the similarity rates of the original abstract of each article and the abstracts extracted manually were calculated with four different metrics. These metrics are BLEU, Rouge-N, Rouge-L, and METEOR. The metrics are calculated by applying each algorithm. For each article, the results of these metrics were averaged, and the average results were determined. After these processes, the algorithm that gave the best results for both methods was determined, and Topic Modelling and Keyword Extraction operations were performed on the algorithm that gave these results. According to these, the topics of the article were determined, and keywords were determined for the determined topics.

Algorithms and approaches in the study are given below in Figure 4. Looking at the literature, an algorithm can belong to more than one approach. In this study, each algorithm was not selected from only one approach, but the average results were determined after determining all algorithms belonging to that approach for each approach.

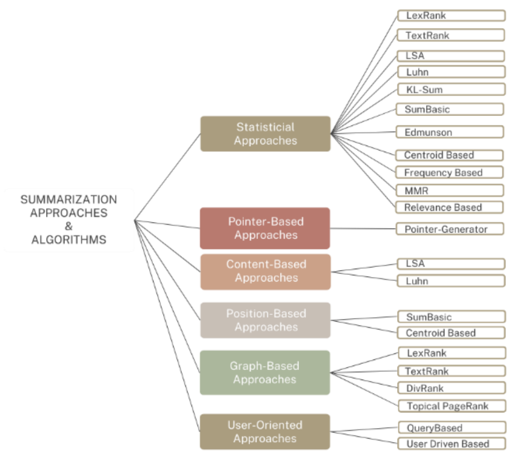

Figure 4. The Approaches and Algorithms used in the study.

Experimental results

This paper provides a comprehensive evaluation of various text summarization algorithms covering statistical, graph-based, content-based, position-based, pointer-based, and user-oriented approaches. These algorithms have been rigorously tested on a dataset of 50 full-text articles covering various topics such as Cyber Security, Artificial Intelligence, Blockchain, Deep Learning, Big Data, IoT, and Image Processing. Two different summarization methods were applied, one covering a significant portion of the articles (from Introduction to References) and the other focusing on a shorter section (from Conclusion to References). The section from the Introduction to the References section is referred to as Method 1, and the section from the Conclusion to the References section is referred to as Method 2. The study aimed to identify the most effective summarization technique through quantitative evaluation.

It was determined in the study that text lengths affect the performance of text summarization while summarizing. Accordingly, the similarity of Abstract to Conclusion drew attention in this study. Therefore, in this study, two methods were considered according to text lengths and their similarity ratios were determined. Accordingly, it is determined that method 2 gives better results in summarization.

The summaries generated by each algorithm were compared to the original full-text article summaries using various evaluation metrics, including ROUGE-N, ROUGE-L, BLEU, and METEOR, to assess overlapping n-grams, longest common sequences, and semantic similarity.

Table 2 shows the results of the similarity metrics mentioned above for each algorithm. The difference between the similarity metrics rises up to 43 percent. In order to eliminate this problem, the study focuses on a single metric, which is the arithmetic mean of all metrics. As mentioned before, after the results of the two methods were determined, the average similarity ratios were determined by taking the arithmetic mean of the similarity metrics, which are given in Table 5.

As indicated in Table 2, a high difference between the metrics indicates that the arithmetic mean will give better results. The results of the metrics of the approaches are also shown in Table 3. In the table where the metric results are given for both methods, there is again a significant difference between the metrics. Therefore, the average results of the approaches are shown in Table 4 by taking the arithmetic mean of the metrics for the approaches.

In the light of the findings, results were obtained for both methods. When these results were analyzed, Content-Based approaches produced the best results when the arithmetic average of the metrics for method 1 in article summarization was taken. The similarity of the summary to the real summary was determined as the highest score here and was determined as 0.328. For Method 2, Content-Based approaches are again in the leading position with 0.343. Based on these results, we can say that Content-Based approaches are better than other approaches for inferential text summarization in scientific articles. We can see the results of all approaches in Table 4 below.

According to the results of Table 4, the Content-Based approach shows a higher success rate than the other methods. The fact that this approach has success rates of 0.328 and 0.343 under Method 1 and Method 2, respectively, shows that this approach is more effective than the others.

According to the results of Table 4. the difference between the best and worst performing approaches is quite significant. While the best result is obtained in the Content-Based approach, the worst result is obtained in the Pointer-Based approach, which has a success rate of only 0.155 under Method 1. The difference between these results is 173 percent, which shows a significant difference between the best and the worst results. This indicates that the Content-Based approach is superior to the others, and the Pointer-Based approach is less effective in the context of this particular problem.

It is conceivable that the LSA and Luhn algorithms within the Content Based approach could play a critical role in achieving this high success. How these algorithms work and what they take into account may require further investigation to explain this high success.

In addition, the algorithm that gives the best results over all the algorithms studied, regardless of the approaches, is determined for Method 1 and Method 2 and shown in Table 4. Taking the arithmetic mean of the metrics, the best algorithm for Method 1 was the LexRank algorithm with 0.349 accuracy, while the best algorithm for Method 2 was the User Driven Based with 0.361 accuracy. The results of all algorithms are shown in Table 5.

Table 2. Metric scores per algorithm.

Metric Scores Per Algorithm	**Method 1**				**Method 2**			
	BLEU	**RougeN**	**RougeL**	**METEOR**	**BLEU**	**RougeN**	**RougeL**	**METEOR**
LexRank	0.293	0.381	0.167	0.556	0.370	0.364	0.185	0.480
TextRank	0.179	0.382	0.141	0.609	0.335	0.362	0.172	0.499
LSA	0.236	0.369	0.143	0.531	0.350	0.356	0.174	0.476
Luhn	0.114	0.410	0.131	0.688	0.313	0.369	0.166	0.521
KL-Sum	0.248	0.330	0.143	0.496	0.365	0.332	0.165	0.443
SumBasic	0.364	0.256	0.129	0.293	0.387	0.315	0.158	0.399
Edmunson	0.189	0.376	0.147	0.597	0.340	0.376	0.187	0.511
Centroid Based	0.282	0.320	0.152	0.502	0.389	0.320	0.158	0.426
Frequency Based	0.439	0.227	0.132	0.258	0.474	0.322	0.185	0.426
MMR	0.315	0.325	0.156	0.465	0.366	0.257	0.130	0.372
Relevance Based	0.345	0.313	0.157	0.436	0.417	0.318	0.168	0.412
PointerGenerator	0.319	0.117	0.064	0.121	0.464	0.315	0.181	0.385
DivRank	0.359	0.284	0.146	0.384	0.433	0.312	0.172	0.382
Topical Pagerank	0.362	0.275	0.142	0.371	0.420	0.283	0.149	0.344
QueryBased	0.409	0.204	0.123	0.230	0.415	0.184	0.122	0.177
User-Driven	0.409	0.315	0.161	0.429	0.486	0.352	0.197	0.445

Table 3. Metric scores per approach.

Metric Scores Per Approach	**Method 1**				**Method 2**			
	BLEU	**RougeN**	**RougeL**	**METEOR**	**BLEU**	**RougeN**	**RougeL**	**METEOR**
Statistical	0.273	0.335	0.145	0.494	0.373	0.336	0.138	0.451
Graph-Based	0.298	0.331	0.149	0.480	0.390	0.330	0.170	0.426
Content-Based	0.175	0.390	0.137	0.610	0.332	0.363	0.170	0.499
Position-Based	0.323	0.288	0.141	0.398	0.388	0.318	0.158	0.413
Pointer-Based	0.319	0.117	0.064	0.121	0.464	0.315	0.181	0.385
User-Oriented	0.409	0.260	0.142	0.330	0.451	0.268	0.160	0.311

Table 4. Results of the approaches.

Approaches	**Method 1**	**Method 2**
Statisticial	0,312	0,329
Content Based	**0,328**	**0,343**
Graph Based	0,314	0,327
Pointer Based	0,155	0,336
Position Based	0,288	0,316
User Oriented	0,285	0,293

Table 5. Results of the algorithms.

Algorithms	**Method 1 results**	**Method 2 results**
Lex_Rank	**0,349**	0,344
Text_Rank	0,328	0,339
LSA	0,319	0,339
Luhn	0,336	0,347
KL-Sum	0,304	0,323
Sum_Basic	0,261	0,314
Edmunson	0,327	0,349
Centroid_Based	0,314	0,318
Frequency-Based	0,264	0,347
MMR	0,314	0,274
Relevance-Based	0,312	0,323
DivRank	0,292	0,325

Topical PageRank	0,286	0,299
Pointer Generator	0,155	0,336
Query Based	0,241	0,225
User Driven Based	0,329	**0,361**

In the study, four metrics were determined for the results of the approaches, and algorithms and the arithmetic average of these metrics constituted our results. On the other hand, when the metric that gives the highest result is accepted instead of the arithmetic average of the metrics, the algorithm and approach change.

In this study, the arithmetic mean was used to evaluate 4 metrics. The aim was to provide a consistent and fair evaluation of the dataset by providing equal weighting for each metric.

The use of a weighted average produces different results for each metric, complicating the evaluation and undermining the reliability of the results. Our observations have shown that the use of weighted averages can distort the overall picture by overestimating or underestimating the importance of some metrics.

With the arithmetic mean, each metric had a 25 per cent share and all metrics affected the system equally. This allowed us to make a clearer and more objective assessment of the dataset.

As a result, the use of the arithmetic mean created an equal and fair evaluation ground for the 4 metrics in our study. Avoiding the weighted average ensured a consistent and reliable analysis of the dataset.

When the arithmetic mean of the metrics and the metrics that produce the highest results are selected, we can see in Table 6 and Table 7 that although the Content-Based approach is the best approach as the arithmetic mean of the metrics, when we try the metric that produces the highest results, Statistical approaches produce the highest results for Method 1 and Method 2.

Table 6. Results of the average metrics and best scores metrics for method 1.

Average Results & Best Results for Method 1		
Approaches	**Average Metric Result**	**Best Metric Result**
Statistical	0,312	**0,389**
Content-Based	**0,328**	0,363
Graph-Based	0,314	0,371
Pointer-Based	0,155	0,155
Position-Based	0,288	0,327
User-Oriented	0,285	0,343

Table 7. Results of the average metrics and best scores metrics for method 2.

Average Results & Best Results for Method 2		
Approaches	**Average Metric Result**	**Best Metric Result**
Statisticial	0,329	**0,397**
Content Based	**0,343**	0,358
Graph Based	0,327	0,366
Pointer Based	0,336	0,336
Position Based	0,316	0,338
User Oriented	0,293	0,373

In addition, the LDA algorithm was used for the keywords to be obtained at the end of the model as the output of the study, and the outputs achieved a success of 0.41 when compared with the keywords of the original text.

As a result, it seems that the best approach, according to different similarity metrics, is the Content-Based approach for extractive text summarization for scientific articles. In addition, it was determined that LexRank is the algorithm that works best in the face of changing ratios as the size of the text increases.

Conclusion

This comprehensive benchmarking of extractive text summarization techniques on scientific documents provides data-driven insights to guide future research. The quantitative advantages of the different algorithms are described. This study provides a recommendation for future research on scientific papers and theses. It will be useful for abstract and keyword extraction in the conversion and publication of theses into articles. Researchers can find more appropriate results by using Abstractive text summarization techniques.

The study's key findings and results can be summarized as follows:

1. Evaluation of Summarization Algorithms: The performance of each summarization algorithm was systematically evaluated using quantitative metrics. For each algorithm, summaries were generated using both the first and second summarization methods, and the results were compared.
2. Identification of the Best-Performing Algorithm: The algorithm that yielded the highest average evaluation scores across the four metrics (ROUGE-N, ROUGE-L, BLEU, METEOR) was considered the best-performing approach. Additionally, the algorithm with the highest individual score for each metric across all algorithms was noted.
3. Optimized Summaries: The top-performing algorithm was employed to generate optimized summaries for each article. These summaries were subjected to Latent Dirichlet Allocation (LDA) topic modeling to identify core themes and keywords.
4. Topic Modeling and Keyword Extraction: The topics and keywords extracted from the optimized summaries were manually reviewed and filtered to gain insights into the main themes and key terms covered in the articles.

In conclusion, this comprehensive benchmark evaluation shed light on the effectiveness of various summarization algorithms across a diverse range of articles. By averaging metric scores, the study identified the top-performing technique, which in turn generated optimized summaries. The subsequent topic modeling and keyword extraction processes provided valuable insights into the core themes and keywords within the articles, further emphasizing the importance of effective summarization techniques in text analysis and understanding.

References

[1] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, “Automatic text summarization: A comprehensive survey,” *Expert Syst Appl*, vol. 165, p. 113679, 2021.

[2] A. Dash, A. Shandilya, A. Biswas, K. Ghosh, S. Ghosh, and A. Chakraborty, “Summarizing user-generated textual content: Motivation and methods for fairness in algorithmic summaries,” *Proc ACM Hum Comput Interact*, vol. 3, no. CSCW, pp. 1–28, 2019.

[3] N. Alami, M. El Mallahi, H. Amakdouf, and H. Qjidaa, “Hybrid method for text summarization based on statistical and semantic treatment,” *Multimed Tools Appl*, vol. 80, pp. 19567–19600, 2021.

[4] A. Kanapala, S. Pal, and R. Pamula, “Text summarization from legal documents: a survey,” *Artif Intell Rev*, vol. 51, pp. 371–402, 2019.

[5] S. Song, H. Huang, and T. Ruan, “Abstractive text summarization using LSTM-CNN based deep learning,” *Multimed Tools Appl*, vol. 78, pp. 857–875, 2019.

[6] T. Liu, “A Hybrid Automatic Text summarization Model for Judgment Documents”.

[7] D. Yadav, J. Desai, and A. K. Yadav, “Automatic text summarization methods: A comprehensive review,” *arXiv preprint arXiv:2204.01849*, 2022.

[8] G. Erkan and D. R. Radev, “Lexrank: Graph-based lexical centrality as salience in text summarization,” *Journal of artificial intelligence research*, vol. 22, pp. 457–479, 2004.

[9] R. Mihalcea and P. Tarau, “A language independent algorithm for single and multiple document summarization,” in *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*, 2005.

[10] C. Mallick, A. K. Das, M. Dutta, A. K. Das, and A. Sarkar, “Graph-based text summarization using modified TextRank,” in *Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018*, Springer, 2019, pp. 137–146.

[11] K. Kireyev, “Using Latent Semantic Analysis for Extractive Summarization.,” in *TAC*, 2008.

[12] K. Srividya, S. K. Bommuluri, V. V. V. K. Asapu, T. R. Illa, V. R. Basa, and R. V. S. Chatradi, “A Hybrid Approach for Automatic Text Summarization and Translation based On Luhn, Pegasus, and Textrank Algorithms,” in *2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, IEEE, 2022, pp. 1–8.

[13] C. Fang, D. Mu, Z. Deng, and Z. Wu, “Word-sentence co-ranking for automatic extractive text summarization,” *Expert Syst Appl*, vol. 72, pp. 189–195, 2017.

[14] V. Gupta and G. S. Lehal, “A survey of text summarization extractive techniques,” *Journal of emerging technologies in web intelligence*, vol. 2, no. 3, pp. 258–268, 2010.

[15] A. Sinha, A. Yadav, and A. Gahlot, “Extractive text summarization using neural networks,” *arXiv preprint arXiv:1802.10137*, 2018.

[16] D. Miller, “Leveraging BERT for extractive text summarization on lectures,” *arXiv preprint arXiv:1906.04165*, 2019.

[17] J. Xu, Z. Gan, Y. Cheng, and J. Liu, “Discourse-aware neural extractive text summarization,” *arXiv preprint arXiv:1910.14142*, 2019.

[18] J. N. Madhuri and R. G. Kumar, “Extractive text summarization using sentence ranking,” in *2019 international conference on data science and communication (IconDSC)*, IEEE, 2019, pp. 1–3.

[19] R. Alguliev and R. Aliguliyev, “Evolutionary algorithm for extractive text summarization,” *Intell Inf Manag*, vol. 1, no. 02, p. 128, 2009.

[20] J. Xu and G. Durrett, “Neural extractive text summarization with syntactic compression,” *arXiv preprint arXiv:1902.00863*, 2019.

[21] N. S. Shirwandkar and S. Kulkarni, “Extractive text summarization using deep learning,” in *2018 fourth international conference on computing communication control and automation (ICCUBEA)*, IEEE, 2018, pp. 1–5.

[22] R. A. García-Hernández and Y. Ledeneva, “Single extractive text summarization based on a genetic algorithm,” in *Pattern Recognition: 5th Mexican Conference, MCPR 2013, Querétaro, Mexico, June 26-29, 2013. Proceedings 5*, Springer, 2013, pp. 374–383.

[23] Q. A. Al-Radaideh and D. Q. Bataineh, “A hybrid approach for arabic text summarization using domain knowledge and genetic algorithms,” *Cognit Comput*, vol. 10, pp. 651–669, 2018.

[24] R. C. Belwal, S. Rai, and A. Gupta, “A new graph-based extractive text summarization using keywords or topic modeling,” *J Ambient Intell Humaniz Comput*, vol. 12, no. 10, pp. 8975–8990, 2021.

[25] R. Rani and D. K. Lobiyal, “An extractive text summarization approach using tagged-LDA based topic modeling,” *Multimed Tools Appl*, vol. 80, pp. 3275–3305, 2021.

[26] J. He, W. Kryściński, B. McCann, N. Rajani, and C. Xiong, “Ctrlsum: Towards generic controllable text summarization,” *arXiv preprint arXiv:2012.04281*, 2020.

[27] M. Mohd, R. Jan, and M. Shah, “Text document summarization using word embedding,” *Expert Syst Appl*, vol. 143, p. 112958, 2020.

[28] D. Wang, P. Liu, M. Zhong, J. Fu, X. Qiu, and X. Huang, “Exploring domain shift in extractive text summarization,” *arXiv preprint arXiv:1908.11664*, 2019.

[29] M. Yousefi-Azar and L. Hamey, “Text summarization using unsupervised deep learning,” *Expert Syst Appl*, vol. 68, pp. 93–105, 2017.

[30] IEEE, “IEEE Xplore.” Accessed: Oct. 12, 2023. [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp

[31] C. Kruengkrai and C. Jaruskulchai, “Generic text summarization using local and global properties of sentences,” in *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*, IEEE, 2003, pp. 201–206.

[32] M. Moradi and N. Ghadiri, “Quantifying the informativeness for biomedical literature summarization: An itemset mining method,” *Comput Methods Programs Biomed*, vol. 146, pp. 77–89, 2017.

[33] Y. Ko and J. Seo, “An effective sentence-extraction technique using contextual information and statistical approaches for text summarization,” *Pattern Recognit Lett*, vol. 29, no. 9, pp. 1366–1371, 2008.

[34] C. Mallick, A. K. Das, M. Dutta, A. K. Das, and A. Sarkar, “Graph-based text summarization using modified TextRank,” in *Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018*, Springer, 2019, pp. 137–146.

[35] R. C. Belwal, S. Rai, and A. Gupta, “A new graph-based extractive text summarization using keywords or topic modeling,” *J Ambient Intell Humaniz Comput*, vol. 12, no. 10, pp. 8975–8990, 2021.

[36] S. Beliga, A. Meštrović, and S. Martinčić-Ipšić, “An overview of graph-based keyword extraction methods and approaches,” *Journal of information and organizational sciences*, vol. 39, no. 1, pp. 1–20, 2015.

[37] O. Sornil and K. Gree-Ut, “An automatic text summarization approach using content-based and graph-based characteristics,” in *2006 IEEE Conference on Cybernetics and Intelligent Systems*, IEEE, 2006, pp. 1–6.

[38] H. M. M. Hasan, F. Sanyal, and D. Chaki, “A novel approach to extract important keywords from documents applying latent semantic analysis,” in *2018 10th International Conference on Knowledge and Smart Technology (KST)*, IEEE, 2018, pp. 117–122.

[39] S. Gholamrezazadeh, M. A. Salehi, and B. Gholamzadeh, “A comprehensive survey on text summarization systems,” in *2009 2nd International Conference on Computer Science and its Applications*, IEEE, 2009, pp. 1–6.

[40] A. S. Schwartz and M. A. Hearst, “A simple algorithm for identifying abbreviation definitions in biomedical text,” in *Biocomputing 2003,* World Scientific, 2002, pp. 451–462.

[41] A. Dash, A. Shandilya, A. Biswas, K. Ghosh, S. Ghosh, and A. Chakraborty, “Summarizing user-generated textual content: Motivation and methods for fairness in algorithmic summaries,” *Proc ACM Hum Comput Interact*, vol. 3, no. CSCW, pp. 1–28, 2019.

[42] N. Elhadad, M.-Y. Kan, J. L. Klavans, and K. R. McKeown, “Customization in a unified framework for summarizing medical literature,” *Artif Intell Med*, vol. 33, no. 2, pp. 179–198, 2005.

[43] M. Song, Y. Feng, and L. Jing, “HISum: Hyperbolic Interaction Model for Extractive Multi-Document Summarization,” in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 1427–1436.

[44] D. Yadav, J. Desai, and A. K. Yadav, “Automatic text summarization methods: A comprehensive review,” *arXiv preprint arXiv:2204.01849*, 2022.

[45] S. Upasani, N. Amin, S. Damania, A. Jadhav, and A. M. Jagtap, “Automatic summary generation using textrank based extractive text summarization technique,” 2020.

[46] D. M. Victor, F. F. Eduardo, R. Biswas, E. Alegre, and L. Fernández-Robles, “Application of extractive text summarization algorithms to speech-to-text media,” in *Hybrid Artificial Intelligent Systems: 14th International Conference, HAIS 2019, León, Spain, September 4–6, 2019, Proceedings 14*, Springer, 2019, pp. 540–550.

[47] S. Sah, S. Kulhare, A. Gray, S. Venugopalan, E. Prud’Hommeaux, and R. Ptucha, “Semantic text summarization of long videos,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV),* IEEE, 2017, pp. 989–997.

[48] G. Rossiello, P. Basile, and G. Semeraro, “Centroid-based text summarization through compositionality of word embeddings,” in *Proceedings of the multiling 2017 workshop on summarization and summary evaluation across source types and genres*, 2017, pp. 12–21.

[49] A. Hajjar and J. Tekli, “Unsupervised extractive text summarization using frequency-based sentence clustering,” in *European Conference on Advances in Databases and Information Systems*, Springer, 2022, pp. 245–255.

[50] P. Verma and H. Om, “Extraction based text summarization methods on user’s review data: A comparative study,” in *Smart Trends in Information Technology and Computer Communications: First International Conference, SmartCom 2016, Jaipur, India, August 6–7, 2016, Revised Selected Papers 1*, Springer, 2016, pp. 346–354.

[51] R. Hao, Y. Li, Y. Feng, and Z. Chen, “Are duplicates really harmful? An empirical study on bug report summarization techniques,” *Journal of Software: Evolution and Process*, p. e2424, 2022.

[52] A. Reunamo *et al.*, “Text Classification Model Explainability for Keyword Extraction–Towards Keyword-Based Summarization of Nursing Care Episodes,” in *MEDINFO 2021: One World, One Health–Global Partnership for Digital Innovation*, IOS Press, 2022, pp. 632–636.

[53] A. Kumar, S. Seth, S. Gupta, and S. Maini, “Sentic computing for aspect-based opinion summarization using multi-head attention with feature pooled pointer generator network,” *Cognit Comput*, vol. 14, no. 1, pp. 130–148, 2022.

[54] N. Rahman and B. Borah, “Improvement of query-based text summarization using word sense disambiguation,” *Complex & Intelligent Systems*, vol. 6, pp. 75–85, 2020.

[55] N. Gu and R. H. R. Hahnloser, “SciLit: A Platform for Joint Scientific Literature Discovery, Summarization and Citation Generation,” *arXiv preprint arXiv:2306.03535,* 2023.

[56] P. Gupta, S. Nigam, and R. Singh, “A Statistical Language Modeling Framework for Extractive Summarization of Text Documents,” *SN Comput Sci*, vol. 4, no. 6, p. 750, 2023.

[57] H. C. Manh, H. Le Thanh, and T. L. Minh, “Extractive Multi-document Summarization using

K-means, Centroid-based Method, MMR, and Sentence Position," in *Proceedings of the 10th International Symposium on Information and Communication Technology*, 2019, pp. 29–35.

[58] R. Parimoo, R. Sharma, N. Gaur, N. Jain, and S. Bansal, "A review on text summarization techniques,"," *Int J Res Appl Sci Eng Technol*, vol. 10, no. 5, pp. 871–873, 2022.

[59] W. Xiao and G. Carenini, "Extractive summarization of long documents by combining global and local context," *arXiv preprint arXiv:1909.08089*, 2019.

[60] N. Giarelis, C. Mastrokostas, and N. Karacapilidis, "Abstractive vs. Extractive Summarization: An Experimental Review," *Applied Sciences*, vol. 13, no. 13, p. 7620, 2023.

[61] A.-N. Dutulescu, M. Dascalu, and S. Ruseti, "Unsupervised Extractive Summarization with BERT," in *2022 24th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, IEEE, 2022, pp. 158–164.