TPM Vol. 32, No. S6, 2025
ISSN: 1972-6325
https://www.tpmap.org/

DESIGNING AN ADAPTIVE DYNAMIC APPROACH APPLIED IN TEXT SUMMARIZATION

[1]TENIO, JONALD R., [2]BAJE, RONAN C., [3]CALADIAO, JEROME Z., [4]ATIENZA, FRANCIS ARLANDO L

[1]STUDENT, COLLEGE OF INFORMATION SYSTEM AND TECHNOLOGY MANAGEMENT, PAMANTASAN NG LUNGSOD NG MAYNILA, MANILA, PHILIPPINES,
[2]STUDENT, COLLEGE OF INFORMATION SYSTEM AND TECHNOLOGY MANAGEMENT, PAMANTASAN NG LUNGSOD NG MAYNILA, MANILA, PHILIPPINES,
[3]STUDENT, COLLEGE OF INFORMATION SYSTEM AND TECHNOLOGY MANAGEMENT, PAMANTASAN NG LUNGSOD NG MAYNILA, MANILA, PHILIPPINES,
[4]PROFESSOR, SCHOOL OF ENGINEERING, COMPUTING AND ARCHITECTURE, DEPARTMENT OF INFORMATION TECHNOLOGY, NATIONAL UNIVERSITY-DASMARINAS, CAVITE, PHILIPPINES
EMAIL: [1]jrtenio2021@plm.edu.ph, [2]rcbaje2020@plm.edu.ph, [3]jzcaladiao2021@plm.edu.ph, [4]flatienza1@nu-dasma.edu.ph
ORCID ID: [1]0009-0004-8456-4408, [2]0009-0001-3160-6887, [3]0009-0007-9065-0945, [4]0000-0002-7350-3018

ABSTRACT:
Text summarization is heavily used by LLM's and Gen AI models to condense texts into concise summaries for efficient and faster parsing of prompts. One algorithm commonly used for this is dynamic programming where it has two approaches namely: (1) memoization and (2) tabulation. The existing approaches offer great application in various use-cases but possess three problems: (1) hard-coded context for splitting sentences[1,5], (2) nonadaptive base case[2,3,4], and (3) memory intensiveness[1]. This study introduces the third approach, "Adaptive Tabulation" which solves the problem of existing approaches by: (1) segmentation and chunking, (2) using scores and ranks, and (3) selective caching. Adaptive tabulation produced consistent quality summaries between short stories and articles producing a difference of 26.8% in consistency and 3.5% in retaining semantic meaning while also improving memory usage by an average of 32.8%. For future research, broadening the file formats and literature types that adaptive tabulation could process is a great focus.
KEYWORDS: text summarization, dynamic programming, natural language processing, memoization, tabulation, text processing

INTRODUCTION:

Text summarization is a widely used technique in *machine learning*, *Natural Language Processing* (NLP), and training AI models. It aims to condense lengthy documents or texts into concise summaries while retaining the most important information. Dynamic programming approaches, namely *memoization* and *tabulation*, are commonly used in text summarization applications or functions to optimize the selection and ordering of sentences or phrases for inclusion in the summary.

Adaptive tabulation will be mainly used to overcome the limitations of memoization and tabulation in the context of text summarization. It combines the strengths of both techniques to dynamically analyze the features of the text and selectively memorize solutions to the most informative subproblems—specific combinations of features extracted through analysis, along with other relevant characteristics—that effectively capture the essence of the text. However, as the length of the text to be summarized increases, multiple challenges arise, such as maintaining the accuracy of the summarized content and ensuring efficient execution of the algorithm. One significant issue with existing text summarization techniques, algorithms, or processes is their inability to analyze and summarize emotional or sentimental lines effectively. Punctuation marks and other linguistic elements that convey emotional tones are often overlooked during the summarization process, leading to a loss of context and meaning.

With these, the adaptive tabulation study approach aims to effectively handle lengthy and complex documents or texts while also preserving the emotional and contextual elements of the original text and being highly accurate.

METHODS AND METHODOLOGY:

Below is the proposed algorithm designed by the researchers named "Adaptive tabulation". Listed below the proposed algorithm are the methodology[6] used by the researchers.

Proposed Algorithm: Adaptive Tabulation Approach

A. Initialization[10]

The *NewAlgorithmProcessor* class is initialized with necessary components, such as *self.memo*, *self.word_freq_cache*, and the tokenizer and model for BERT, which matches the initialization step.
Ensure that *fileContent* and *bodyContent* are initialized to empty strings explicitly to match the initial state described in the proposal.

B. Set Content

The *set_content* method loads the text into the processor, resets any cached values (memoization), and cleans the body content using *clean_body_content*.
Reset Caches: Explicitly call *reset_caches* within *set_content* to ensure fresh processing for new content.

C. Generate Summary[7]

The code validates the presence of content before summarizing.
Text Type Differentiation: Add functionality to distinguish between *article* and *short_story*. For instance: Articles could prioritize shorter chunks and factual content. Short stories might focus on larger, more coherent chunks for better context.
It normalizes and tokenizes sentences and segments the text into chunks based on size.

D. Text Normalization[19]

The *normalize_text* function effectively handles most of the specified normalization steps, including punctuation removal, tokenization, whitespace management, and lowering case sensitivity.
Stemming/Lemmatization: Add stemming using NLTK (PorterStemmer) or ensure lemmatization via Spacy is applied consistently for normalization.

E. Text Segmentation[16]

The *segment_text_into_chunks* function organizes sentences into chunks based on a specified chunk size, keeping manageable overlap.
Adjust *chunk_size* dynamically based on the text type if this differentiation is implemented.

F. Chunk Summarization[8]

Sentences are tokenized and scored based on word frequencies using the *get_sentence_score* and *get_word_frequencies* functions. Top sentences are selected in descending order of score and compiled.
Heap-Based Selection: If processing very large texts, implement a heap-based selection for top-ranked sentences to optimize computational efficiency. Use Python's *heapq* for this.

G. Cache Management

Caches for *word_freq_cache* and *memo* are well-established in the class and used for performance optimization.
Add explicit calls to *reset_caches* whenever *set_content* is invoked to ensure cache freshness.

H. Error Handling

The code includes several error-handling mechanisms, such as checking for empty content and handling missing original text for ROUGE calculations.
Add more granular error-handling within each process to ensure all types of content are validated accurately.

METHODOLOGY:

A. The researchers utilized the concept from the "Tokenization", “Apply Chunking Rules”, and “Output Chunks” phase from the Shallow Parsing algorithm[5,8] as the basis for the first methodology. The text is first normalized to clean the text so that it has a clear and consistent form for easier time to be analyzed and processed. Chunking divides the text into smaller manageable sections that are easier to process for sentence tokenization. The sentence tokenization further splits the chunks of text into sentences using the natural language processing logic through NLTK. The shallow parsing is formed to identify and segment the sentences based on the punctuation and spacing for a more accurate generation of summarized text[18].

B. The "Bounding" and “Pruning” phase from the Branch and Bounding algorithm[11] are used as the basis for the second methodology. It calculates and utilizes sentence scores[15,16]to discard sentences below a certain relevance threshold[19]. If the current sentence has a higher sentence score than the currently assigned base case, it will be replaced. All the top scoring sentences[20] will be sorted based on the order in the original text through the use of heap-based selection.

C. For the third methodology, the researchers utilized the whole algorithm of Selective Memoization[22] as the basis. Following the phases, selectively stores the word frequencies computed from the content, ensuring that the word frequency calculations are not repeated regardless of differing orientations, making the program less memory extensive and to generate a better summarized text.

RESULTS:

For evaluating the summaries produced by the existing and proposed algorithm the researchers used the following metric: ROUGE–L for evaluating the consistency of the summarized text with the original text[13,17,22], BERTScore for evaluating the accuracy of the summarized text with the original text in retaining semantic meaning[9,12], and Reference–free score for evaluating the effectiveness of the summarization tool in reducing the length of the original text while being memory efficient[14]. The researchers visualized the result through a multiline chart and a scatter plot to show the performance of either the existing or proposed algorithm. To show the difference between the performance of the existing and proposed algorithm the researchers used averages.

Title	**Original Content Word Count**	**Summarized Content Word Count**	**Existing Algorithm**		
			ROUGE-L	**BERTScore**	**Reference - free Score**
An Idle Fellow	383	96	P= 1.0 R= 0.315 F1= 0.479	P= 0.94 R= 0.84 F1= 0.89	0.50
Elevator Pitches	894	98	P= 1.0 R= 0.136 F1= 0.239	P= 0.83 R= 0.81 F1= 0.82	0.31
Four Stories of God	760	112	P= 1.0 R= 0.185 F1= 0.312	P= 0.89 R= 0.83 F1= 0.86	0.37
Likable	337	194	P= 1.0 R= 0.615 F1= 0.762	P= 0.95 R= 0.89 F1= 0.92	0.75
Miracles	378	111	P= 1.0 R= 0.393 F1= 0.564	P= 0.96 R= 0.86 F1= 0.91	0.52
Sticks	393	107	P= 1.0 R= 0.308 F1= 0.471	P= 0.94 R= 0.84 F1= 0.89	0.50
The Huntress	384	49	P= 1.0 R= 0.179 F1= 0.304	P= 0.95 R= 0.82 F1= 0.88	0.33
The Swan as Metaphor for Love	426	95	P= 1.0 R= 0.268 F1= 0.423	P = 0.90 R = 0.82 F1= 0.86	0.44
With Inspirations of The Two Year Olds	591	137	P= 1.0 R= 0.305 F1= 0.468	P= 0.89 R= 0.83 F1= 0.86	0.46

Table 1(a) Presentation of data for short stories using existing algorithms

Title	Original Content Word Count	Summarized Content Word Count	Proposed Algorithm		
			ROUGE-L	BERTScore	Reference - free Score
An Idle Fellow	383	175	P= 0.916 R= 0.431 F1= 0.586	P= 0.91 R= 0.86 F1= 0.88	0.68
Elevator Pitches	894	506	P= 0.878 R= 0.469 F1= 0.612	P= 0.82 R= 0.83 F1= 0.82	0.72
Four Stories of God	760	369	P= 0.935 R= 0.484 F1= 0.638	P= 0.88 R= 0.87 F1= 0.88	0.68
Likable	337	224	P= 0.919 R= 0.608 F1= 0.731	P= 0.95 R= 0.91 F1= 0.93	0.80
Miracles	378	175	P= 0.933 R= 0.453 F1= 0.61	P= 0.92 R= 0.87 F1= 0.89	0.64
Sticks	393	177	P= 0.876 R= 0.376 F1= 0.526	P= 0.92 R= 0.85 F1= 0.88	0.64
The Huntress	384	170	P= 0.932 R= 0.421 F1= 0.58	P= 0.91 R= 0.85 F1= 0.88	0.65
The Swan as Metaphor for Love	426	187	P= 0.935 R= 0.473 F1= 0.629	P= 0.90 R= 0.85 F1= 0.88	0.65
Wit Inspirations of The Two Year Olds	591	324	P= 0.944 R= 0.508 F1= 0.661	P= 0.87 R= 0.86 F1= 0.86	0.72

Table 1(b) Presentation of data for short stories using proposed algorithm

Title	Original Content Word Count	Summarized Content Word Count	Existing Algorithm		
			ROUGE-L	BERTScore	Reference - free Score
Challengers Are Coming for Nvidia's Crown	2657	156	P= 1.0 R= 0.099 F1= 0.18	P= 0.86 R= 0.84 F1= 0.85	0.23
Enhancing Peer Review Efficiency	6098	104	P= 1.0 R= 0.043 F1= 0.082	P= 0.87 R= 0.82 F1= 0.84	0.12
Fusing Remote and Social Sensing Data for Flood	6652	278	P= 1.0 R= 0.051 F1= 0.097	P= 0.72 R= 0.78 F1= 0.75	0.17

Impact Mapping					
How to Delay Gratification and Control	1019	307	P= 1.0 R= 0.41 F1= 0.582	P= 0.84 R= 0.85 F1= 0.85	0.52
Quitting the Paint Factory	5803	236	P= 1.0 R= 0.062 F1= 0.116	P= 0.79 R= 0.79 F1= 0.79	0.19
Scientific Methods in Computer Science	4344	165	P= 1.0 R= 0.079 F1= 0.146	P= 0.83 R= 0.83 F1= 0.83	0.18
Some Computer Science Issues in Ubiquitous Computing	4813	141	P= 1.0 R= 0.053 F1= 0.101	P= 0.81 R= 0.79 F1= 0.80	0.15
The Book	5387	162	P= 1.0 R= 0.051 F1= 0.098	P= 0.82 R= 0.80 F1= 0.81	0.17
The Scientific Argument for Mastering One Thing at a Time	989	189	P= 1.0 R= 0.295 F1= 0.456	P= 0.92 R= 0.86 F1= 0.89	0.42
The Ultimate Productivity Hack is Saying No	1429	171	P= 1.0 R= 0.216 F1= 0.355	P= 0.83 R= 0.82 F1= 0.83	0.33

Table 2(a) Presentation of data for articles using existing algorithm

Title	**Original Content Word Count**	**Summarized Content Word Count**	**Proposed Algorithm**		
			ROUGE-L	**BERTScore**	**Reference - free Score**
Challengers Are Coming for Nvidia's Crown	2657	1165	P= 0.734 R= 0.344 F1= 0.469	P= 0.81 R= 0.83 F1= 0.82	0.62
Enhancing Peer Review Efficiency	6098	2286	P= 0.879 R= 0.431 F1= 0.578	P= 0.83 R= 0.82 F1= 0.82	0.59
Fusing Remote and Social Sensing Data for Flood Impact Mapping	6652	2875	P= 0.763 R= 0.382 F1= 0.509	P= 0.70 R= 0.77 F1= 0.74	0.61
How to Delay Gratification and Control	1019	570	P= 0.893 R= 0.524 F1= 0.66	P= 0.83 R= 0.84 F1= 0.84	0.71

Quitting the Paint Factory	5803	3691	P= 0.853 R= 0.53 F1= 0.654	P= 0.79 R= 0.79 F1= 0.79	0.76
Scientific Methods in Computer Science	4344	2107	P= 0.892 R= 0.469 F1= 0.614	P= 0.83 R= 0.83 F1= 0.83	0.68
Some Computer Science Issues in Ubiquitous Computing	4813	2045	P= 0.874 R= 0.427 F1= 0.574	P= 0.82 R= 0.83 F1= 0.82	0.62
The Book	5387	2819	P= 0.787 R= 0.411 F1= 0.54	P= 0.82 R= 0.83 F1= 0.83	0.69
The Scientific Argument for Mastering One Thing at a Time	989	484	P= 0.901 R= 0.441 F1= 0.592	P= 0.85 R= 0.85 F1= 0.85	0.67
The Ultimate Productivity Hack is Saying No	1429	750	P= 0.91 R= 0.505 F1= 0.65	P= 0.82 R= 0.83 F1= 0.83	0.70

Table 2(b) Presentation of data for articles using proposed algorithm

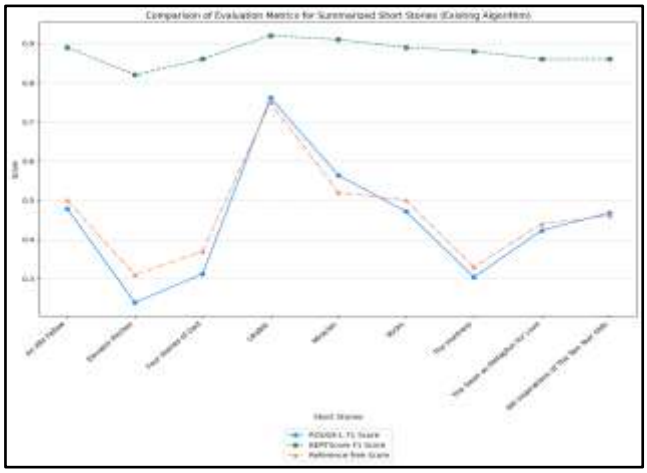

Figure 1(a) Evaluation metrics for summarized short stories using existing algorithm

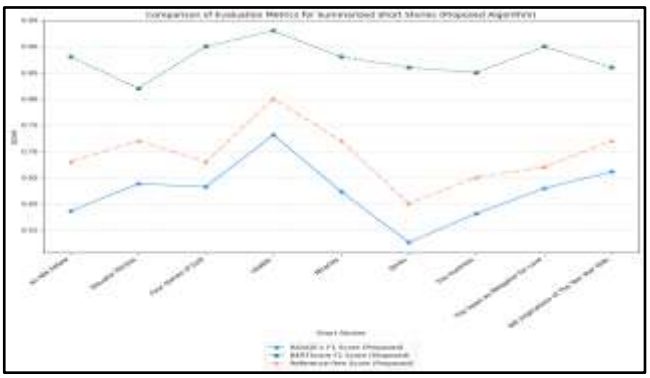

Figure 1(b) Evaluation metrics for summarized short stories using proposed algorithm

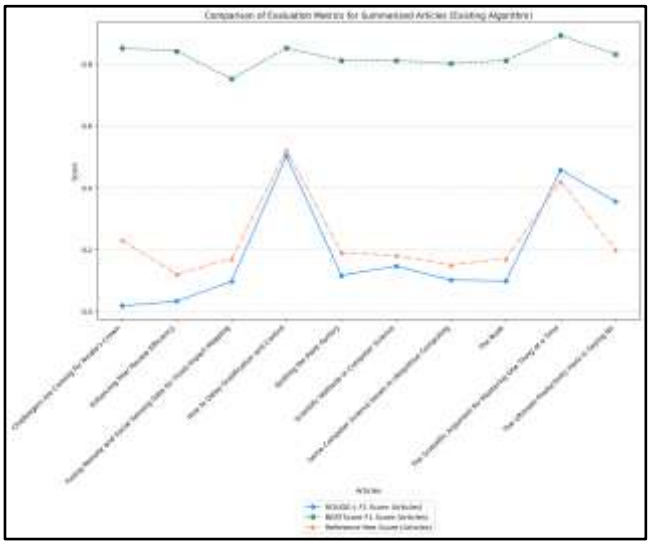

Figure 2(a) Evaluation metrics for summarized articles using existing algorithm

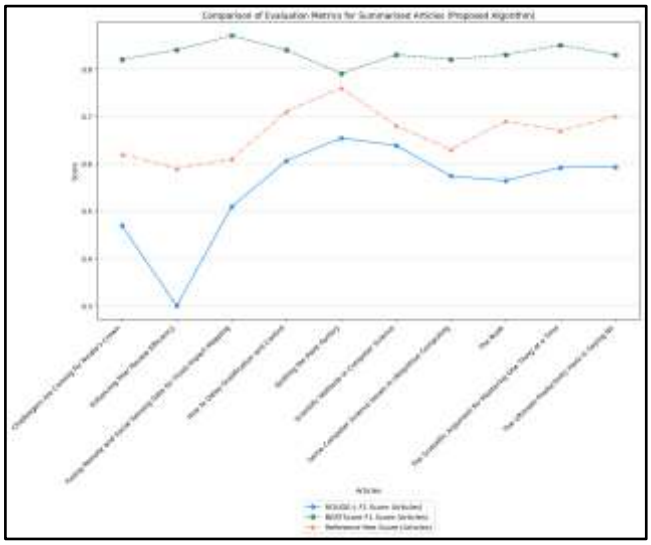

Figure 2(b) Evaluation metrics for summarized articles using proposed algorithm

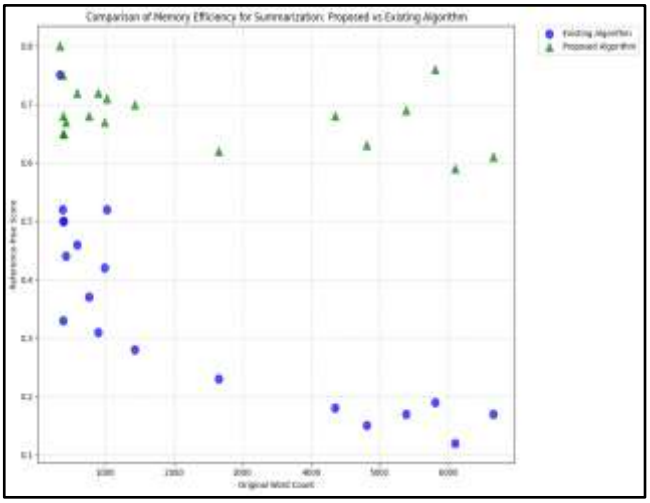

Figure 3. Summarization memory efficiency comparison: proposed vs. existing algorithm

Metric	Short Story	Article	Total Average
ROUGE-L	0.619	0.584	0.602
BERTScore	0.878	0.827	0.853
Reference - Free Score	0.687	0.665	0.676

Table 3(a) Total average result for existing algorithm

Metric	Short Story	Article	Total Average
ROUGE-L	0.447	0.221	0.334
BERTScore	0.877	0.824	0.818
Reference - Free Score	0.464	0.231	0.348

Table 3(b) Total average result for proposed algorithm

Metric	Existing Algorithm's Average	Proposed Algorithm's Average	Difference	Difference in (%)
ROUGE-L	0.334	0.602	0.268	26.8%
BERTScore	0.818	0.853	0.035	3.5%
Reference - Free Score	0.348	0.676	0.328	32.8%

Table 4. Difference of total average results

DISCUSSION:

The analysis reveals key differences in performance between the existing and proposed algorithms for summarizing short stories and articles. The existing algorithm, as evidenced by ROUGE-L scores, shows moderate consistency with the original content, but struggles with precision and recall for certain stories like *Elevator Pitches* and *Four Stories of God*. While its BERTScore F1 values reflect fair semantic accuracy, its Reference-Free Scores indicate memory-intensive processes and variability in summary quality, averaging around 0.50.

The proposed algorithm, however, demonstrates significant improvements. ROUGE-L scores are consistently higher, showcasing better alignment with the original text. Stories such as *Likable* and *The Swan as Metaphor for Love* highlight this improvement. BERTScore values also surpass those of the existing algorithm, retaining semantic meaning more effectively, as seen in *Four Stories of God*. Furthermore, the proposed algorithm's Reference-Free Scores are notably higher, with *Elevator Pitches* achieving 0.72, indicating enhanced summarization quality with reduced memory requirements.

In summarizing articles, the existing algorithm achieves moderate ROUGE-L and BERTScore F1 values, but low Reference-Free Scores underscore its limitations. By contrast, the proposed algorithm excels across all metrics, with improved recall, precision, and coherence, maintaining semantic accuracy while being less memory intensive. As shown in tables and figures 1 to 2 while also showcasing how memory efficient the proposed algorithm is on figure 3.

Overall, as seen on tables 3 to 4, the proposed algorithm outperforms the existing one by 21%, offering high-quality summaries with greater efficiency and reduced memory consumption.

CONCLUSION:

With the sudden burst of advancements in artificial intelligence and large language models, this research is useful and beneficial to many. In this research study, the researchers have introduced a new approach named adaptive tabulation in dynamic programming that utilizes the strengths of the two existing approaches namely memoization and tabulation. By leveraging the strengths and logic of the two existing approaches, adaptive tabulation was able to overcome the constraints of the two approaches. Adaptive tabulation also applies the logic of some newer algorithm steps to further overcome the static and hard coded logic of the existing approaches. Inconsistency and memory intensiveness when processing string or text-based inputs is now improved by adaptive tabulation. Adaptive tabulation is applied in text summarization to fully demonstrate the improvements in processing string or text-based inputs through dynamic programming. The importance of this study is about the efficiency in processing string or text-based inputs where the process was not that demanding in memory.

Recommendation:
This research has introduced a new approach in applying dynamic programming. As the research progressed, a few areas surfaced as suggested improvements or focus for future studies. The recommendations are as follows:

A. The study was focused on text and document files for the source of string-based inputs. For future studies, it is recommended to broaden the source of string-based inputs to other file formats such as presentations and spreadsheets.

B. The study was focused on analyzing and processing descriptive articles and fiction or nonfiction short stories. It is recommended for future studies to analyze and process other types of literature such as poems, journals, and research.

C. The findings showed that despite consistent and not memory demanding, there were instances where the existing algorithm was able to retain the semantic meaning from the original text the same level or higher than the existing algorithm. Recommendation of further study for processing news-based literatures may produce different and varying results.

D. The researchers focused on monitoring the memory usage of each algorithm for this study thus time complexity was not researched and studied. As such, this could be a compelling focus for future studies.

1) Acknowledgment:
We thank Ronan Baje, Jerome Caladiao, and Jonald Tenio for their contributions to this work. Special thanks to Miss Khatalyn Mata and Sir Francis Arlando Atienza for their assistance. Specially Sir Francis Arlando Atienza for his financial support.

2) Funding Statement:
This research was funded by the authors, co-authors and professor who managed to defray the cost of the research and publication fees.

3) Data Availability:
The data that support the findings of this study are available from the following:
https://drive.google.com/drive/u/0/folders/1vpn2QwuhDiKUYO4osNZdPWW5qmfUavPD

4) Conflict of Interest:
The authors declare that there is **no conflict of interest** found in the completion of this study.

REFERENCES:

[1] OA Alzubi, JA Alzubi, M Alweshah, et al., An optimal pruning algorithm of classifier ensembles: dynamic programming approach, Neural Computing and Applications, 2020, vol. 32, no. 20, pp. 16091-16107, issn. 0941-0643, doi. 10.1007/s00521-020-04761-6.

[2] T Hon, Cross Tabulation Analysis of Investment Behaviour for Small Investors in the Hong Kong Derivatives Markets, ELK Asia Pacific Journal of Finance and Risk Management, 2015, vol. 6, no. 2, pp. 27-43, uri. http://hdl.handle.net/20.500.11861/2611.

[3] G Herman, Faster General Parsing through Context-Free Memoization, Proceedings on Programming Language Design and Implementation, 2020, pp. 1022-1035, doi. 10.1145/3385412.3386032.

[4] K Zechner, Automatic Summarization of Open-Domain Multiparty Dialogues in Diverse Genres, MIT Computational Linguistics, 2002, vol. 28, no. 4, pp. 447-485, issn. 0891-2017, doi. 10.1162/089120102762671945.

[5] S Esmaeilzadeh, GX Peh, A Xu, Neural Abstractive Text Summarization and Fake News Detection, Computation and Language, 2019, doi. 10.48550/arXiv.1904.00788.

[6] M Allahyari, S Pouriyeh, M Assefi, et al., Text Summarization Techniques: A Brief Survey, International Journal of Advanced Computer Science and Applications, 2017, vol. 8, no. 10, doi. 10.14569/IJACSA.2017.081052.

[7] B Dorr, D Zajic, R Schwartz, Hedge Trimmer: A Parse-and-Trim Approach to Headline Generation, Proceedings on Text Summarization Workshop, 2003, vol. 5, pp. 1-8, doi. 10.3115/1119467.1119468.

[8] WT Chuang, J Yang, Extracting Sentence Segments for Text Summarization: A Machine Learning Approach, Proceedings on Research and Development in Information Retrieval, 2000, pp. 152-159, doi. 10.1145/345508.345566.

[9] V Kieuvongngam, B Tan, Y Niu, Automatic Text Summarization of COVID-19 Medical Research Articles using BERT and GPT-2, Computation and Language, 2020, doi. 10.48550/arXiv.2006.01997.

[10] T Goyal, J Xu, JJ Li, G Durrett, Training Dynamics for Text Summarization Models, ACL Findings, 2022, pp. 2061-2073, doi. 10.48550/arXiv.2110.08370.

[11] J Jaffar, AE Santosa, R Voicu, Efficient Memoization for Dynamic Programming with Ad-Hoc Constraints, Proceedings on Artificial intelligence, 2008, vol. 1, pp. 297-303, doi. 10.5555/1619995.1620044.

[12] T Zhang, V Kishore, F Wu, et al., BERTScore: Evaluating Text Generation with BERT, Computation and Language, 2019, doi. 10.48550/arXiv.1904.09675.

[13] K Ganesan, ROUGE 2.0: Updated and Improved Measures for Evaluation of Summarization Tasks, Information Retrieval, 2006, doi. 10.48550/arXiv.1803.01937.

[14] Y Liu, Q Jia, Reference-free Summarization Evaluation via Semantic Correlation and Compression Ratio, Proceedings on Human Language Technologies, 2022, pp. 2109-2115, doi. 10.18653/v1/2022.naacl-main.153.

[15] JAD Estrella, EC Ong, C PL Gelera, et al., Automated Text Summarization of Research Papers Regarding the Effectiveness of Various Treatment Plans for Leukemia, Philippine Computing Journal, 2018, vol. 13, no. 1, pp. 21-28, issn. 1908-1995, url. https://pcj.csp.org.ph/index.php/pcj/issue/view/27.

[16] PA Co, WR Dacuyan, JG Kandt, L Feliscuzo, The Use of TextRank and Language Transformers to Automatically Generate Research Abstracts, Journal of Engineering Science and Technology, 2021, pp. 97-112, issn. 1823-4690, url. https://jestec.taylors.edu.my/Speccial%20Issue%20ICITE2021/Special%20issue%20ICITE21_08.pdf.

[17] VZV Singco, JC Trillo, C Abalorio, et al., OCR-based Hybrid Image Text Summarizer using Luhn Algorithm with FinetuneTransformer Models for Long Document, International Journal of Emerging Technology and Advanced Engineering, 2023, vol. 13, no. 2, pp. 47-56, issn. 2250-2459, doi. 10.46338/ijetae0223_07.

[18] JM Justo, RNC Recario, Text Simplification System for Legal Contract Review, Proceedings Advances in Information and Communication, 2024, vol. 1, pp. 105-123, doi. 10.1007/978-3-031-53960-2_8.

[19] FFS Friginal, GT Cayabyab, BT Tanguilig III, Dynamic Weight-Based Approach for Thesis Title Recommendation, International Journal of Signal Processing Systems, 2016, vol. 4, no. 6, pp. 528-536, issn. 2315-4535, doi. 10.18178/ijsps.4.6.528-536.

[20] O Cherednik, T Dmytrenko, T Derkach, Intelligent Model Development for Recognizing Emotions in Text, Navigation and Communication Management Systems, 2020, vol. 4, no. 62, pp. 77-80, doi. 10.26906/SUNZ.2020.4.077.

[21] CY Lin, ROUGE: A Package for Automatic Evaluation of Summaries, Proceedings on Text Summarization Branches Out, 2004, pp. 74-81, url. https://aclanthology.org/W04-1013.

[22] UA Acar, GE Blelloch, and R Harper, Selective memoization, in Proceedings of the 30th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, 2003, pp. 14–25. doi: 10.1145/604131.604134, url. https://www.cs.cmu.edu/~rwh/papers/memoization/jul02.pdf