Iraqi Journal for Computers and Informatics
Available online: https://ijci.uoitc.edu.iq/index.php/ijci
Volume 52, issue 1. 2026

Print ISSN: 2313-190X, Online ISSN: 2520-4912

An Efficient Model for Academic Paper Title Generation using Summarization Approach

Ali Ahmed Ali [a], Mohammed Ali Mohammed [b]

[a] *Informatics Institute for Postgraduate Studies, University of Information Technology and Communications (UOITC), Baghdad, Iraq.*
[b] *Business Informatics College, University of Information Technology and Communications (UOITC), Baghdad, Iraq.*

ARTICLEINFO

Keywords:

Carefully select an appropriate list of five keywords that represents the real content of your paper

ABSTRACT

The title should concisely encapsulate the overall content; however, producing a strong academic title that reflects the core contribution remains challenging. This paper proposes and evaluates a pipeline-based model for academic paper title generation: abstracts collected from the NIPS dataset are preprocessed, condensed via extractive summarization, and then Standard YAKE is applied to extract weighted keywords. Titles are generated from a fixed Top-6 keyword budget while preserving the YAKE ranking order; in this work, the proposed model refers to the end-to-end pipeline configuration rather than a newly trained neural architecture. We compare four summarization algorithms (Luhn, LSA, Edmundson, and KL) based on their YAKE-weighted keywords, and adopt Luhn as default because it produces more topic-relevant YAKE-weighted keywords than the other summarizers. We illustrate the pipeline by using three qualitative examples and compare three generator-Ateeqq/keywords-title-generator, KeyToText (k2t), and GPT-2 (gpt2)-under identical Top-6 constraints. Experiments on 300 papers using both lexical and semantic similarity metrics (ROUGE-L, TF-IDF cosine similarity, BERTScore-F1, and SciBERTScore-F1) indicate that the adopted Luhn→YAKE→Top-6→LLM pipeline produces the most semantically aligned titles under identical input constraints.

1. INTRODUCTION

Recent studies highlight the importance of research-paper titles as the first point of contact for readers and reviewers. Titles influence initial screening and perceived relevance, as well as visibility and readership, which can affect citation impact [1], [2], [3], [4]. However, many authors devote insufficient attention to careful title selection [2].

A title should provide a concise, informative summary of the paper's topic and contribution [1]. This motivates automated title generation methods that exploit textual signals from the paper itself. In particular, extractive summarization has been widely used in the news domain to condense documents for downstream tasks [5], [6], [7]. However, a significant part of the methods remains based on selecting relevant keywords to compose a title without a sufficient model of the structure and purpose of the paper [8]. In spite of the advancement in the field of text generation, high quality academic titles are still difficult to generate. The complexity is due to the different fields of science, inconsistent styles of writing and the necessity to summarize the main idea in a few words. Moreover, an assessment of title quality cannot be performed without complexity: many automatic evaluation metrics emphasize lexical overlap and grammatical structure but fail to represent the underlying aims and purpose of

E-mail address:
ms202410015@iips.edu.iq [a]
mohammed.ali@uoitc.edu.iq [b]

Corresponding* : *Ali Ahmed Ali*

Received 10 December 2025, Accepted 6 March 2026

DOI: 10.25195/ijci.v52i1701.

the paper [9]. These issues drive a regulated, repeatable title generation and assessment pipeline. This study addresses this gap through the presentation of a controlled pipeline that sequentially connects summarization, YAKE keyword weighting and keyword-to-title generation with constant Top-6 keyword budget, allowing individual components to be easily compared. The goal of this study is to evaluate how four extractive summarization algorithms (Luhn, Edmundson, LSA, and KL) affect YAKE weighting of keywords, and compare three keyword-to-title generators (Large Language Model (LLM)-based, KeyToText, and GPT-2) on a fixed keyword budget (Top-6) by applying lexical (ROUGE-L, TF-IDF) and semantic (BERTScore, SciBERTScore) metrics. The paper is organized as follows: Section 2 is a review of related work on academic title generation. The proposed pipeline is described in section 3. The results and discussion are provided in Section 4. Finally, Section 5 concludes the paper.

2. RELATED WORK

Title generation (Academic title generation) has also received a lot of interest in the studies of text generation. Putra and Khodra [10] (2017) introduced a summarization-based title generation system, which is used to generate several candidate titles as the scientific article titles, with regard to the linguistic structure and textual units.

Gu et al. [11] in 2020 designed a headline-generation model that had been trained on a large news corpus (21,190 items)

tested on a held-out test set (1,006 items). They compared generated headlines to human written references with the help of F1-based measures and word-order constraints.

Bajaj et al. [12] [2022] suggested a title generation method (TiGen) using transformers to generate titles of scholarly texts.

Lodhwal and Choudhary [13] (2023) focused on automatic title generation on the basis of recurrent neural networks and pre-trained Transformer language models, which claim positive results on arXiv-style data with promising results. Their findings highlight the effectiveness of neural models in capturing informative cues for concise title generation.

Rehman et al. [14] (2024) compared various pre-trained language models in title generation in research papers, such as GPT - 3.5-turbo and PEGASUS-large. They found that the choice of model can significantly affect lexical and semantic correspondence with reference titles, as measured by ROUGE, METEOR, BERTScore, and SciBERTScore.

3. METHODOLOGY

This study aims to design and implement a pipeline to generate academic paper titles. To enhance the efficiency and reproducibility of the pipeline, the pipeline incorporates preprocessing, extractive summarization, and keyword extraction, as well as, key word-to-title generation. Instead of introducing a new neural architecture, this study evaluates existing components under a controlled configuration. The proposed pipeline has the following workflow as shown in figure 1.

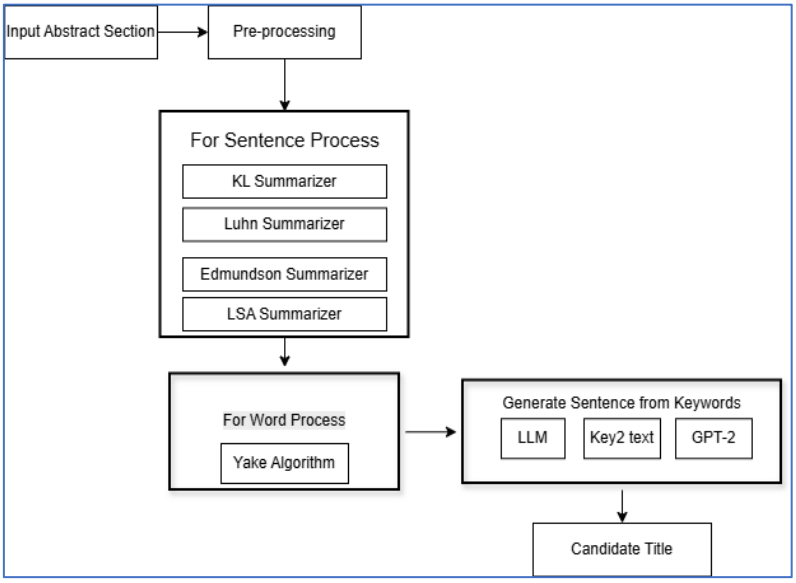

Figure 1. *Flowchart of the Proposed Model*

In preprocessing stage, most documents and texts have unnecessary content such as stop words, slang expressions, and misspellings. Such irrelevant parts cause noise, which may adversely affect the performance of a range of algorithms, in particular, those that rely on probabilistic and statistical learning procedures [15], [16]. The abstract is selected as the input because the abstract presents the main concept of the research in a summary form [17]. Text preprocessing and filtering are also important in text-mining and sentiment-analysis tasks because they reduce noisy terms and improve the quality of downstream textual representations [18], [19], [20].

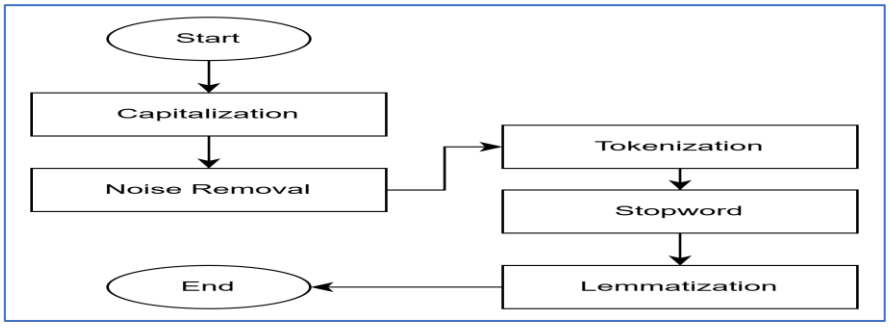

Figure 2. *Flowchart of the Preprocessing Model*

The abstract is condensed into a representative set of sentences by using extractive summarization, which has been widely applied to reduce long documents into shorter representative forms for downstream processing [21]. We consider four summarizers (Luhn, Edmundson, LSA, and KL) to study how the summarization choice affects downstream keyword weighting and title generation. In brief, Luhn scores sentences by using term frequency [22]; Edmundson extends frequency-based scoring with cue/bonus/stigma words [23]; LSA selects salient sentences by capturing global contextual relationships [24]; and KL-based summarization selects sentences that best preserve the original term-distribution [25]. After summarization, we apply the YAKE algorithm to extract weighted keywords from the condensed text [26]. YAKE is an unsupervised keyword extraction method that ranks candidates by using statistical features such as term position, frequency, casing, and context; lower YAKE scores indicate higher importance. For analysis, we report the Top-10 keywords, while for title generation, we use only the Top-6 keywords, preserving the YAKE ranking order, as the generator input. The fixed Top-6 keyword budget was selected to keep the generator input concise, fair, and comparable across all generators. The first six YAKE-ranked terms usually capture the main topic, method, and context needed for an academic title, whereas lower-ranked terms may add noise or force the generator to include secondary details. Therefore, the Top-10 keywords are reported for transparency, but only the first six keywords, in the original YAKE order, are used for title generation. For the Large Language Model (LLM)-based title generator, we use the publicly available Ateeqq/keywords-title-generator model, which is a T5-base sequence-to-sequence model fine-tuned for keyword-to-title generation.

To make the evaluation procedure more explicit, each generated title G is compared with its corresponding reference title R using lexical and semantic similarity metrics. First, TF-IDF cosine similarity is used to measure the vector-space similarity between the generated and reference titles:

$$\text{Cosine}(G, R) = (VG \cdot VR) / (||VG||\ ||VR||) \qquad \textbf{Equation (1)}$$

where VG and VR represent the TF-IDF vectors of the generated and reference titles, respectively.

ROUGE-L is used to measure lexical overlap based on the longest common subsequence between G and R:

$$RLCS = LCS(G, R) / |R|, \quad PLCS = LCS(G, R) / |G| \qquad \textbf{Equation (2)}$$

$$FLCS = ((1 + \beta^2) \times RLCS \times PLCS) / (RLCS + \beta^2 \times PLCS) \qquad \textbf{Equation (3)}$$

where RLCS and PLCS denote recall and precision based on the longest common subsequence.

For keyword extraction, YAKE assigns each candidate keyword a score based on local statistical features such as position, frequency, casing, and context. This can be generally represented as:

$$Score(k) = f(position, frequency, casing, context) \qquad \textbf{Equation (4)}$$

where k is a candidate keyword, and a lower YAKE score indicates a more important keyword.

For semantic evaluation, BERTScore and SciBERTScore compare generated and reference titles using contextual embeddings rather than exact word overlap. BERTScore can be generally represented as:

$$\text{BERTScore}(G, R) = F1(Pemb, Remb) \quad \textbf{Equation (5)}$$

where Pemb and Remb are computed from embedding-based token similarities between the generated and reference titles.

4. RESULTS AND DISCUSSION

To illustrate the pipeline, we first summarize three qualitative examples in Table 1. We then present YAKE keyword weights for the illustrative examples under different summarizers (Tables 2-5). Finally, we present an aggregate evaluation on 300 papers with lexical and semantic metrics (Table 7).

Table 1. *Illustrative examples: reference titles, Top-6 keywords, generated titles, and evaluation scores*

Reference title	Paper 1: Data Security and Privacy in Cloud Computing	Paper 2: Learning to Play the Game of Chess	Paper 3: Unsupervised Learning by Convex and Conic Coding
Keywords (Top-6)	Security, privacy, computing, cloud, data, protection	Present, program, games, neurochess, learns, chess	Unsupervised, proposed, convex, conic, encoders, learning
Generated title by LLM	Data Protection and Security in the Cloud: How to Protect Your Data	How to program neurochess games to learn from the present and the past	Unsupervised learning with conic and convex encoders proposed
SciBERT-F1	0.783	0.651	0.853
BERTScore-F1	0.918	0.893	0.895
ROUGE-L	0.421	0.200	0.400
Generated title by Key to text	Cloud Data Security and Privacy Protection	The present| program is games| neurochess| learnings.	Unsupervised learning with convex encoders.
SciBERT-F1	0.705	0.560	0.566
BERTScore-F1	0.887	0.839	0.855
ROUGE-L	0.210	0.286	0.333
Generated title by GPT-2	security privacy computing cloud data protection. [truncated output]	present program games neurochess learns chess. [truncated output]	unsupervised proposed convex conic encoders learning. [truncated output]
SciBERT-F1	0.614	0.622	0.649
BERTScore-F1	0.860	0.868	0.843
ROUGE-L	0.250	0.273	0.300

Discussion (Illustrative examples): The three examples show that the LLM generator produces more coherent academic titles than KeyToText and GPT-2 under the same Top-6 keyword input. KeyToText is rather frequently prone to producing template-like or fragmented results whereas GPT-2 is more likely to repeat keywords or come up with irrelevant continuation text. This motivates the use of semantic measures alongside lexical overlap. We use four extractive summarizers (Luhn, Edmundson, LSA,

and KL) to analyze the influence of summarization on keyword weighting, followed by the application of YAKE on the summaries (Table 2 to Table 5) to get weighted keywords. The Top-10 keywords are reported for analysis, while the Top-6 keywords (in YAKE order) are used as the common input for title generation as shown in Table 1. In the experimental setting, Luhn is taken to be the default summarizer because, based on the keyword-weighting patterns in Tables 2-5 and the aggregate evaluation in Table 7, it produces more topic-relevant YAKE-weighted keywords than the other summarizers.

Table 2. *YAKE Top-10 keywords (Luhn summarizer)*

No.	Paper 1: Data Security and Privacy in Cloud Computing		Paper 2: Learning to Play the Game of Chess		Paper 3: Unsupervised Learning by Convex and Conic Coding	
	Word	Weights	Word	Weights	Word	Weights
1	Security	0.0738	Present	0.1867	unsupervised	0.1689
2	Privacy	0.0766	Program	0.1867	proposed	0.1689
3	Computing	0.0900	Games	0.1867	convex	0.1769
4	Cloud	0.0937	neurochess	0.2619	conic	0.1769
5	Data	0.0962	Learns	0.2619	encoders	0.1812
6	Protection	0.1101	Chess	0.2619	learning	0.2213
7	Industry	0.1163	Learning	0.2788	algorithm	0.2213
8	Business	0.1163	Play	0.2899	based	0.2410
9	Industries	0.1593	Final	0.2899	basis	0.2490
10	Government	0.1593	outcome	0.2899	vector	0.2490

Table 3. *YAKE Top-10 keywords (Edmundson summarizer)*

No.	Paper 1: Data Security and Privacy in Cloud Computing		Paper 2: Learning to Play the Game of Chess		Paper 3: Unsupervised Learning by Convex and Conic Coding	
	Word	Weights	Word	Weights	Word	Weights
1	Security	0.0931	present	0.1733	unsupervised	0.1771
2	Data	0.1057	program	0.1733	proposed	0.1771
3	Technology	0.1586	Games	0.1733	convex	0.2087
4	Privacy	0.1853	neurochess	0.2506	conic	0.2087
5	Cloud	0.1866	Learns	0.2506	encoders	0.2539
6	Consistently	0.2135	Chess	0.2506	learning	0.2768
7	Major	0.2135	Play	0.2715	algorithm	0.2768
8	Issue	0.2135	Final	0.2715	based	0.2768
9	Information	0.2135	outcome	0.2715	input	0.3257
10	Protection	0.2225	functions	0.4214	neural	0.5214

Table 4. *YAKE Top-10 keywords (LSA summarizer)*

No.	Paper 1: Data Security and Privacy in Cloud Computing		Paper 2: Learning to Play the Game of Chess		Paper 3: Unsupervised Learning by Convex and Conic Coding	
	Word	Weights	Word	Weights	Word	Weights
1	Privacy	0.0856	present	0.1733	learning	0.1698
2	Security	0.0857	program	0.1733	encoders	0.1698
3	Data	0.1123	games	0.1733	algorithm	0.2475
4	Protection	0.1268	neurochess	0.2506	vector	0.2475
5	Cloud	0.1350	learns	0.2506	produce	0.2666
6	Technology	0.1716	chess	0.2506	basis	0.2666
7	Main	0.2297	play	0.2715	minimize	0.2666
8	Factor	0.2297	final	0.2715	error	0.2666

9	User	0.2297	outcome	0.2715	reconstruction	0.2934
10	Concern	0.2297	functions	0.4214	analysis	0.4153

Table 5. *YAKE Top-10 keywords (KL summarizer)*

No.	Paper 1: Data Security and Privacy in Cloud Computing		Paper 2: Learning to Play the Game of Chess		Paper 3: Unsupervised Learning by Convex and Conic Coding	
	Word	Weights	Word	Weights	Word	Weights
1	Data	0.0656	neurochess	0.1725	conic	0.1530
2	Security	0.1184	functions	0.1725	convex	0.1638
3	Technology	0.1418	represented	0.1725	unsupervised	0.1726
4	Privacy	0.2088	neural	0.2048	proposed	0.1726
5	Consistently	0.2484	learning	0.2175	algorithm	0.1955
6	Major	0.2484	learns	0.2704	input	0.2071
7	Information	0.2484	chess	0.2704	encoders	0.2167
8	Protection	0.2540	board	0.2704	learning	0.2458
9	Issue	0.2647	evaluation	0.2704	based	0.2458
10	Cloud	0.2921	artificial	0.2704	features	0.5134

Discussion (Summarizer effect): The summarization option rearranges priorities of YAKE keywords affecting directly the generator input. In the adopted experimental setting, Luhn tends to promote topic-defining words (smaller YAKE scores) that are consistent with the intention of the reference title, whereas alternatives (e.g., KL) can shift the Top-6 list toward less informative words. Hence, we use Luhn as the default summarizer as it gives higher levels of topic-relevant YAKE-weighted keywords to feed downstream title generation. As reported in Tables 2-5, every summarization algorithm gives a list of ranked YAKE keywords; some core terms in the list can be common to all summarizers, though weights and rank can vary. In the case of the cloud-security, the words security, privacy, computing, cloud, data and protection get the most priority in the Luhn + Standard YAKE setting, which implies a high level of relevance to the reference title.

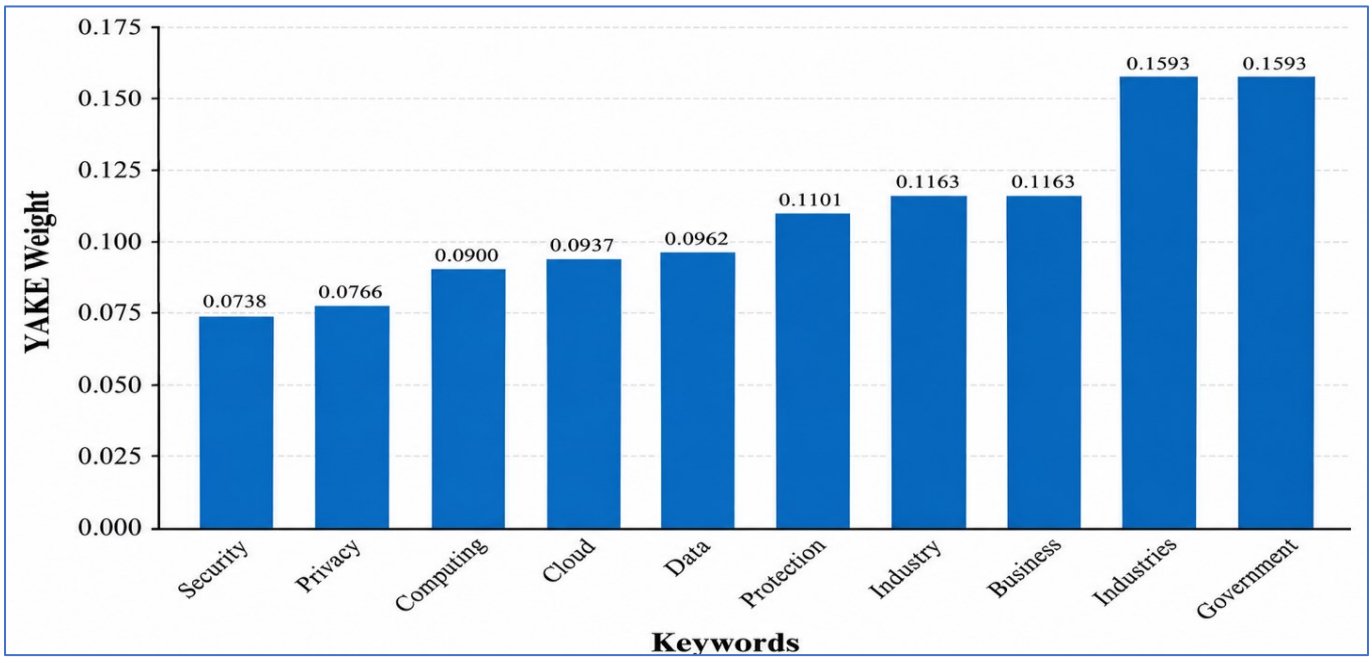

Figure 3. *YAKE Top-10 Keywords and Weights for Paper 1 (Luhn Summarizer)*

In paper 2, Luhn maintains the Top-6 as the central topic keywords (e.g., "neurochess, chess, learns, games), generating clean generator input (although with slight differences in rank with alternative summarizers). In paper 3, Luhn is more focused on the

defining methodological keywords (unsupervised, convex, conic, encoders), in contrast to other summarizers, which can promote other implementation keywords, which substantiates Luhn as the most robust in the adopted Top-6 generation context.

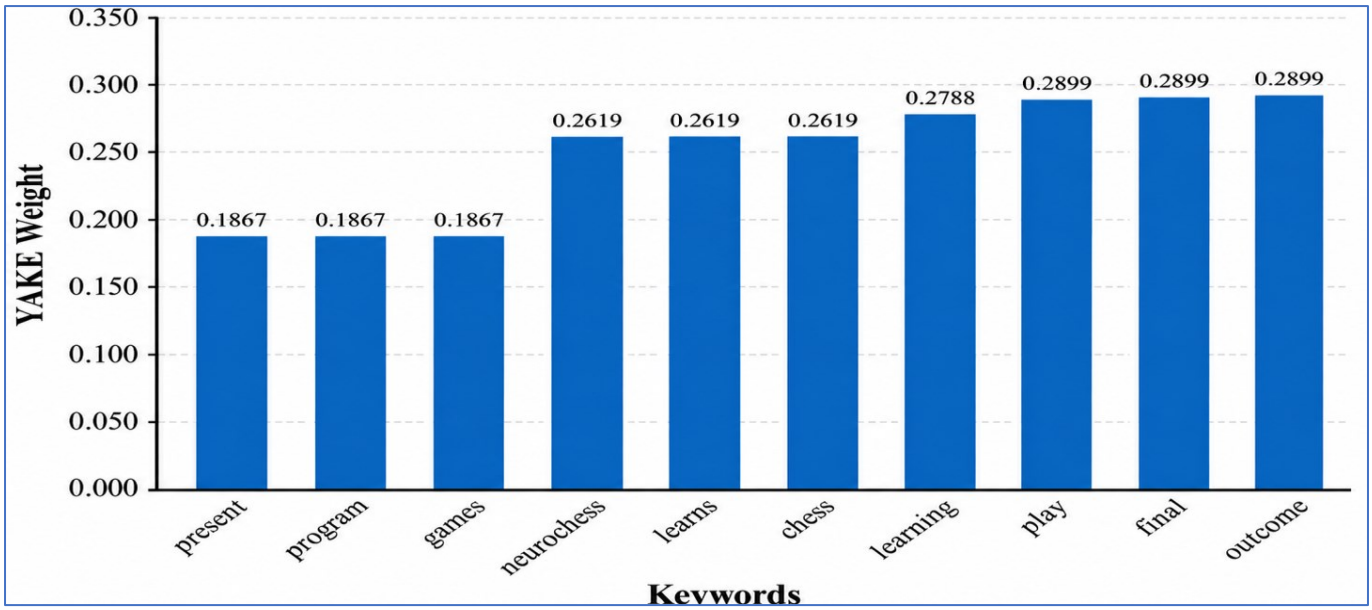

Figure 4. *YAKE Top-10 Keywords and Weights for Paper 2 (Luhn Summarizer)*

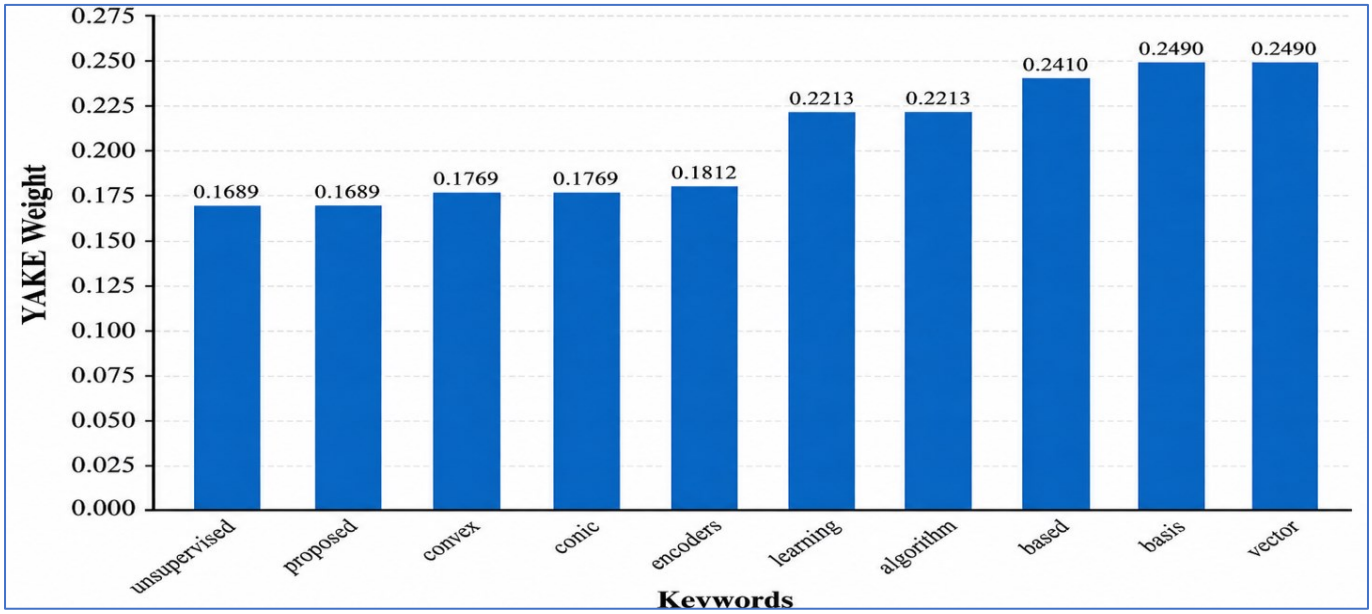

Figure 5*. YAKE Top-10 Keywords and Weights for Paper 3 (Luhn Summarizer)*

Table 6*. Average weighted keyword coverage by summarizer on 700 papers*

Summarizer	Avg. Weighted Coverage	Std. Dev.
Full Abstract (No Summarization)	52.25%	±24.17%
Luhn	53.55%	±24.00%
Edmundson	52.25%	±24.17%
LSA	43.93%	±26.42%

KL-Sum	43.60%	±26.84%

The coverage results in Table 6 show that all candidate inputs (the full abstract and the summaries produced by Luhn, Edmundson, LSA, and KL-Sum) were evaluated using the same YAKE-based keyword-coverage procedure. Among them, Luhn achieved the highest average weighted coverage and was the only summarization method that exceeded the full-abstract baseline. This means that Luhn does not merely shorten the abstract; it tends to retain and emphasize the abstract sentences that contain the most title-relevant terms. Therefore, Luhn is selected as the summarization component in the final pipeline, while the full abstract and the remaining summarizers serve as baselines for comparing the effect of summarization on keyword coverage.

Finally, we generate titles from the Top-6 YAKE keywords using three generators: (i) an LLM-based keyword-to-title generator, (ii) KeyToText, and (iii) GPT-2. All generators receive the same Top-6 keywords (preserving YAKE order). The qualitative outputs for the three illustrative papers are summarized in Table 1, while Table 7 reports the aggregate evaluation on 300 papers using lexical and semantic similarity metrics. The models are identified by their public checkpoints: Ateeqq/keywords-title-generator (LLM-based), KeyToText (k2t), and GPT-2 (gpt2).

Cosine similarity is a metric that measures the similarity between two texts by comparing their vector representations [27], [28]. The cosine similarity result is a value between 0 and 1, where a value of 0 indicates no similarity and a value of 1 indicates identical documents [28], [29]. In contrast, Recall-Oriented Understudy for Gisting Evaluation ROUGE metrics are widely used for summarization evaluation. ROUGE-1 F1 measures the harmonic mean of precision and recall for unigram overlaps, while ROUGE-L F1 evaluates the Longest Common Subsequence LCS between the generated and reference texts, reflecting both sequence order and informatics [30].

Semantic similarity is assessed by using contextual embedding-based metrics [31]. BERTScore is a similarity measure operating at the term level based on contextual embedding of large language models, and typically aligns better with the human judgment than n-gram based metrics [32]. SciBERTScore is grounded on the scientific and technical scientific papers-trained SciBERT, and thus it is more suitable in assessing scholarly text in scientific fields that are related to computer science [33]. These metrics are supplementary to each other, together with TF-IDF cosine similarity and ROUGE, to demonstrate quality and relevance of titles.

For formal clarity, TF-IDF cosine similarity is computed as the cosine of the generated-title and reference-title TF-IDF vectors. ROUGE-L is computed from the longest common subsequence between the generated and reference titles. BERTScore and SciBERTScore compute token-level semantic similarity using contextual embeddings, with SciBERTScore using a scientific-domain encoder. YAKE ranks candidate keywords using local statistical features, where lower scores indicate higher keyword importance.

The illustrative examples are discussed in the order of paper 1, paper 2 and paper 3 so as to achieve uniformity. All the examples are represented in the same experimental setting (Luhn + Standard YAKE + Top-6 keywords in YAKE order) and later an aggregate analysis is conducted on 300 papers (Table 7).

Table 7. *Results on 300 Papers (Mean ± SD and Win Counts)*

Model	N	Semantic F1 (BERTScore / SciBERTScore)	ROUGE-L (mean±std)	TF-IDF cosine (mean±std)	Wins (SciBERT / ROUGE-L) (out of 300)
LLM	300	0.8526±0.0205 / 0.6324±0.0589	0.1755±0.1175	0.1210±0.1080	276/300 (92.0%) / 183/300 (61.0%)
GPT-2	300	0.8377±0.0205 / 0.5576±0.0462	0.1417±0.0958	0.1053±0.0933	6/300 (2.0%) / 48/300 (16.0%)
KeyToText	300	0.8318±0.0188 / 0.5580±0.0400	0.1284±0.1015	0.0823±0.0887	18/300 (6.0%) / 69/300 (23.0%)

Discussion (300-paper evaluation): The difference in the SciBERT win rate (92.0%), and the ROUGE-L win rate (61.0%) indicates that the primary benefit of the adopted Luhn→YAKE→Top-6→LLM is semantic matching and not literal lexical overlap. ROUGE-L rewards surface word-order overlap and penalizes paraphrasing, so it can remain modest even when a title preserves the same meaning. In contrast, BERTScore and SciBERTScore compare contextual meaning, and SciBERT is trained on scientific text, making it more appropriate for evaluating academic titles; and this explains why semantic scores are consistently higher than ROUGE. The keyword-to-title generator is designed to generate coherent titles from a limited keyword budget, whereas GPT-2 is a general language model and KeyToText often produces template-like fragments; therefore, under identical Top-6 constraints, the LLM generator more reliably converts keywords into title.

Limitations**:** The proposed pipeline has several limitations. First, the results depend on the dataset, because abstracts from different domains or writing styles may produce different keyword distributions and title-generation results. Second, the fixed Top-6 keyword budget may not be optimal for all papers: fewer keywords may miss important concepts, while more keywords may introduce less relevant terms. Third, the selected generator may have its own tendency toward certain title structures or wording patterns. Regarding evaluation, lexical metrics such as ROUGE-L and TF-IDF cosine similarity have limited ability to capture meaning when the generated title uses different wording from the reference title. Therefore, they are not used alone; semantic metrics such as BERTScore and SciBERTScore are reported alongside them to provide a more balanced evaluation.

5. CONCLUSION

In conclusion, the summarization algorithm plays an important role because it changes the YAKE keyword weights and, consequently, the generated title quality. Luhn is selected as the default summarizer because it gives the best keyword coverage and is the only summarizer that performs better than using the full abstract. This suggests that Luhn keeps the most important title-related information while removing less useful text. The Top-6 keyword setting is used as a balanced choice: it keeps enough important keywords to represent the paper topic, while avoiding too many keywords that may add noise to the generator. Using the same Top-6 setting for all generators also makes the comparison fair and consistent. Based on the 300-paper evaluation, the Luhn→YAKE→Top-6→LLM pipeline achieves the best semantic similarity results; therefore, it is adopted as the preferred configuration in this study.

Conflicts of Interest

The authors declare no conflicts of interest.

REFERENCES

[1] H. R. Jamali and M. Nikzad, "Article title type and its relation with the number of downloads and citations," Scientometrics, vol. 88, pp. 653–661, 2011, doi: 10.1007/s11192-011-0412-z.

[2] H. Xu, E. Martin, and A. Mahidadia, "Extractive summarization based on keyword profile and language model," in Proc. NAACL-HLT, Denver, CO, USA, 2015, pp. 123–132, doi: 10.3115/v1/N15-1013.

[3] C. E. Paiva, J. P. S. N. Lima, and B. S. R. Paiva, "Articles with short titles describing the results are cited more often," Clinics, vol. 67, no. 5, pp. 509–513, 2012, doi: 10.6061/clinics/2012(05)17.

[4] A. Letchford, H. S. Moat, and T. Preis, "The advantage of short paper titles," R. Soc. Open Sci., vol. 2, no. 8, 2015, Art. no. 150266, doi: 10.1098/rsos.150266.

[5] S. C. Chen and L. S. Lee, "Automatic title generation for Chinese spoken documents using an adaptive k-nearest neighbor approach," in Proc. EUROSPEECH, Geneva, Switzerland, 2003, pp. 2813–2816, doi: 10.21437/Eurospeech.2003-749.

[6] S. Teufel, Argumentative Zoning: Information Extraction from Scientific Text. Ph.D. dissertation, Univ. Edinburgh, Edinburgh, U.K., 1999. [Online]. Available: https://www.cl.cam.ac.uk/~sht25/thesis/t.pdf

[7] W. Li and J. Zhao, "TextRank algorithm by exploiting Wikipedia for short text keywords extraction," in Proc. Int. Conf. Inf. Sci. Control Eng. (ICISCE), Beijing, China, 2016, pp. 683–686, doi: 10.1109/ICISCE.2016.151.

[8] J. W. G. Putra and M. L. Khodra, "Automatic title generation in scientific articles for authorship assistance: A summarization approach," J. ICT Res. Appl., vol. 11, no. 3, pp. 253–267, 2017, doi: 10.5614/itbj.ict.res.appl.2017.11.3.3.

[9] J. Becker, J. P. Wahle, B. Gipp, and T. Ruas, "Text generation: A systematic literature review of tasks, evaluation, and challenges," arXiv preprint arXiv:2405.15604, 2024, doi: 10.48550/arXiv.2405.15604.

[10] A. Waheed, M. Goyal, N. Mittal, and D. Gupta, "Domain controlled title generation with human evaluation," arXiv preprint arXiv:2103.05069, 2021, doi: 10.48550/arXiv.2103.05069.

[11] X. Gu et al., "Generating representative headlines for news stories," in Proc. Web Conf., Taipei, Taiwan, 2020, pp. 1773–1784, doi: 10.1145/3366423.3380247.

[12] D. Bajaj et al., "TiGen-title generator based on deep NLP transformer model for scholarly literature," in Proc. Int. Conf. Commun., Netw. Comput., Singapore, 2022, pp. 297–309, doi: 10.1007/978-3-031-43140-1_26.

[13] V. Lodhwal and G. Choudhary, "Automatic title generation for text with RNN and pre-trained transformer language model," ADBU J. Eng. Technol., vol. 12, no. 2, 2023. [Online]. Available: https://www.researchgate.net/publication/374615369

[14] T. Rehman, D. K. Sanyal, and S. Chattopadhyay, "Can pre-trained language models generate titles for research papers?" in Proc. Int. Conf. Asian Digit. Libraries, Singapore, Dec. 2024, pp. 154–170, doi: 10.1007/978-981-96-0865-2_13.

[15] K. Kowsari et al., "Text classification algorithms: A survey," Information, vol. 10, no. 4, 2019, Art. no. 150, doi: 10.3390/info10040150.

[16] C. C. Aggarwal, Machine Learning for Text. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-73531-3.

[17] J. W. G. Putra and M. L. Khodra, "Rhetorical sentence classification for automatic title generation in scientific article," TELKOMNIKA, vol. 15, no. 2, pp. 656–664, 2017, doi: 10.12928/telkomnika.v15i2.4061.

[18] R. R. Tated and M. M. Ghonge, "A survey on text mining-techniques and application," Int. J. Res. Advent Technol., vol. 1, pp. 380–385, 2015. [Online]. Available: https://ijrat.org/downloads/Conference_Proceedings/icatest2015/ICATEST2015133.pdf

[19] B. Pahwa, S. Taruna, and N. Kasliwal, "Sentiment analysis-strategy for text preprocessing," Int. J. Comput. Appl., vol. 180, no. 34, pp. 15–18, 2018, doi: 10.5120/ijca2018916865.

[20] H. Saif, M. Fernandez, Y. He, and H. Alani, "On stopwords, filtering and data sparsity for sentiment analysis of Twitter," in Proc. Int. Conf. Lang. Resour. Eval. (LREC), Reykjavik, Iceland, 2014, pp. 810–817.

[21] K. Agrawal, "Legal case summarization: An application for text summarization," in Proc. Int. Conf. Comput. Commun. Informatics (ICCCI), Coimbatore, India, 2020, pp. 1–6, doi: 10.1109/ICCCI48352.2020.9104093.

[22] H. P. Luhn, "The automatic creation of literature abstracts," IBM J. Res. Dev., vol. 2, no. 2, pp. 159–165, 1958, doi: 10.1147/rd.22.0159.

[23] H. P. Edmundson, "New methods in automatic extracting," J. ACM, vol. 16, no. 2, pp. 264–285, 1969, doi: 10.1145/321510.321519.

[24] O. Dokun and E. Celebi, "Single document summarization using latent semantic analysis," Int. J. Sci. Res. Inf. Syst. Eng., vol. 1, no. 2, pp. 57–64, 2015. [Online]. Available: https://www.airitilibrary.com/Article/Detail/P20160130001-201512-201601300006-201601300006-57-64

[25] D. Jain, M. D. Borah, and A. Biswas, "Improving Kullback–Leibler based legal document summarization using enhanced text representation," in Proc. IEEE Silchar Subsection Conf. (SILCON), Silchar, India, Nov. 2022, pp. 1–5, doi: 10.1109/SILCON55242.2022.10028887.

[26] R. Campos et al., "YAKE! Collection-independent automatic keyword extractor," in Advances in Information Retrieval, Cham, Switzerland: Springer, 2018, pp. 806–810, doi: 10.1007/978-3-319-76941-7_80.

[27] K. Park, J. S. Hong, and W. Kim, "A methodology combining cosine similarity with classifier for text classification," Appl. Artif. Intell., vol. 34, no. 5, pp. 396–411, 2020, doi: 10.1080/08839514.2020.1723868.

[28] B. I. Kwak, M. L. Han, and H. K. Kim, "Cosine similarity-based anomaly detection methodology for the CAN bus," Expert Syst. Appl., vol. 166, 2021, Art. no. 114066, doi: 10.1016/j.eswa.2020.114066.

[29] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL), Philadelphia, PA, USA, Jul. 2002, pp. 311–318, doi: 10.3115/1073083.1073135.

[30] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in Proc. Workshop Text Summarization Branches Out, Barcelona, Spain, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013.pdf

[31] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT networks," in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2019, pp. 3982–3992, doi: 10.18653/v1/D19-1410.

[32] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating text generation with BERT," in Proc. Int. Conf. Learn. Representations (ICLR), 2020. [Online]. Available: https://arxiv.org/abs/1904.09675

[33] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in Proc. Conf. Empirical Methods Natural Lang. Process. Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 3615–3620, doi: 10.18653/v1/D19-1371.