Knowledge Engineering and Data Science (KEDS)
Vol 6, No 1, April 2023, pp. 57–68

pISSN 2597-4602
eISSN 2597-4637

Maximum Marginal Relevance and Vector Space Model for Summarizing Students' Final Project Abstracts

Gunawan [a,1,*], Fitria [a,2], Esther Irawati Setiawan [a,3], Kimiya Fujisawa [b,3]

[a] *Institut Sains dan Teknologi Terpadu Surabaya,*
Jl. Ngagel Jaya Tengah No.73-77, Surabaya 60284, Indonesia
[b] *Tokyo University of Technology,*
1404-1 Katakuramachi, Hachioji City, Tokyo 192-0982, Japan
[1]*gunawan@stts.edu*;* [2] *fitriatahir@gmail.com*; [3] *esther@stts.edu;* [4] *fujisawa@stf.teu.ac.jp*
** corresponding author*

ARTICLE INFO

Article history:
Received 13 June 2023
Revised 03 July 2023
Accepted 28 July 2023
Published online 31 July 2023

Keywords:
Summary
Query Relevance
Sentence Similarity
Maximum Marginal Relevance

ABSTRACT

Automatic summarization is reducing a text document with a computer program to create a summary that retains the essential parts of the original document. Automatic summarization is necessary to deal with information overload, and the amount of data is increasing. A summary is needed to get the contents of the article briefly. A summary is an effective way to present extended information in a concise form of the main contents of an article, and the aim is to tell the reader the essence of a central idea. The simple concept of a summary is to take an essential part of the entire contents of the article. Which then presents it back in summary form. The steps in this research will start with the user selecting or searching for text documents that will be summarized with keywords in the abstract as a query. The proposed approach performs text preprocessing for documents: sentence breaking, case folding, word tokenizing, filtering, and stemming. The results of the preprocessed text are weighted by term frequency-inverse document frequency (tf-idf), then weighted for query relevance using the vector space model and sentence similarity using cosine similarity. The next stage is maximum marginal relevance for sentence extraction. The proposed approach provides comprehensive summarization compared with another approach. The test results are compared with manual summaries, which produce an average precision of 88%, recall of 61%, and f-measure of 70%.

I. Introduction

A summary represents the article's overview and conveys essential ideas to the reader [1]. Automatic text summarization reduces a text document with a computer program to create a summary that retains the essential parts of the original document [2][3]. The amount of data is increasing to deal with information overload, so automatic summarization is necessary [4]. Summary automation can be applied to single-multi documents and languages [5]. Therefore, an automatic summarizer may ease people in summarizing the data from the web page [6][7], as in the final project and thesis abstract [8].

Maximum marginal relevance (MMR) is an extractive summary method that is used to summarize a single document or multiple documents [9][10]. MMR summarizes documents by calculating the similarity between parts of the text [11][12]. The document segmentation process is carried out in sentences summarizing documents using the MMR method. MMR combines the cosine similarity matrix and VSM to rank sentences in response to the query [13][14]. Most modern information retrieval (IR) search engines produce ranking lists of documents as measured by decreasing relevance to user queries [15][16]. The first assessment to measure the relevant summary results is to measure the relationship between the information in the document and the query given by the user and add the linear combination as a matrix. This linear combination is called marginal relevance [17].

https://doi.org/10.17977/um018v6i12023p57-68

A collaborative initiative to collect and unify existing resources for Indonesian languages, including opening access to previously non-public resources [18]. The paper describes the datasets and standardized data loaders that were brought together through this initiative and discusses the quality of the datasets, which were assessed manually and automatically. We compared the performance of our approach with a summarization initiative from NusaCrowd.

This article consists of four sections. The introduction and context are covered within the first section. The second section describes the research method. The fourth segment describes the results and discussion, while the final section summarizes the conclusions.

II. Method

This research summarizes a document and generates its abstract using an automatic summary system [19][20]. The stages in this research are preprocessing, tf-idf weighting, weighting query relevance, sentence similarity weighting, and MMR for summary extraction [21], as displayed in Figure 1 [22][23].

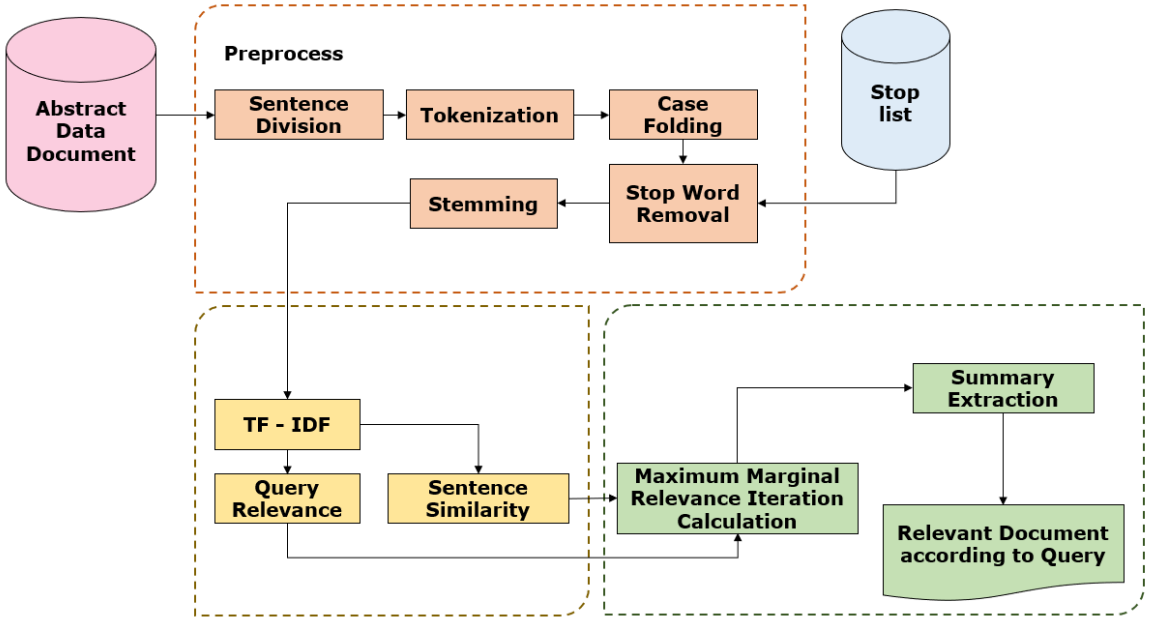

Fig. 1. System architecture

The abstract documents are generally preprocessed (sentence splitting, tokenization, case folding, stopword, and stemming). After preprocessing, tf-idf weighting is carried out, namely, automatic weighting based on the number of occurrences of a word in a document (term frequency) and the number of occurrences in the document collection (inverse document frequency) [24]. The tf-idf weights and calculates the query relevance and sentence similarity weights for weighting query relevance using the vector space model and sentence similarity using cosine similarity [25]. The calculation of the query relevance weight is the weight of the results of comparing the similarity between queries (keywords) to the entire document. At the same time, the sentence similarity weight is the weight of the results of comparing similarities between documents. The next stage of iterative calculations uses maximum marginal relevance by comparing query relevance and sentence similarity to obtain summary extraction to determine the relevant document as a summary [26].

The first step in the text preprocessing stage. is sentence division breaking down documents into sentences. Sentence splitting is breaking long document text strings into a collection of sentences. In breaking the document into sentences using the split () function, with a period ".", question mark "?" and an exclamation point “!” as a delimiter. The sentence splitting stage breaks the document string into a collection of sentences by removing the end of the sentence marks (delimiters). From the results

of sentence splitting, the following steps are tokenization, case folding, stop words removal, stemming, tf-idf weighting, VSM, cosine similarity, and MMR to obtain a summary.

Tokenization is cutting or separating a row of words in a sentence, paragraph, or page into tokens or single-word chunks. This stage also removes certain characters in the form of punctuation marks. Splitting sentences into single words is done by scanning sentences with white space delimiters (spaces, tabs, and newlines).

Case folding is a text processing process in which all text is converted into the same case; in this case, the text is represented in all lowercase letters. The orthographic model error will be corrected by changing all letters to lowercase or lowercase. The following is an example of implementing case folding in summarization [27][28].

Stop words can be referred to as unimportant words, for example, "in", "by", "on", "a", "because", and so on [29][30]. Stop words are removed to remove words that have no connection with documents contained in the database. Examples of other stop words are there, is, is, while, somewhat, he, I, how, and others. Stemming removes a word's prefix or suffix to get the basic word form. For example, registered words and registrations share a common term, stem list [31].

The weighting is obtained based on the number of occurrences of a term in a tf document and the number of occurrences in the idf document collection [32]. The more frequently a word appears in a document, the greater its weight and the smaller it appears in many documents. To calculate the tf-idf weight, use the formula in (1) and (2). Weighting can be obtained based on the number of occurrences of a term in a term frequency (tf) document and the number of occurrences of a term in the inverse document frequency (idf) document collection. The idf value of a term can be calculated as in (3).

$$IDF = Log\left(\frac{d}{df}i\right) \quad (1)$$

d is the number of documents containing the term (t), and dfi is the number of term occurrences against d. The algorithm is used to calculate the weight (W) of each document against keywords (queries).

$$Wd,t = tfd,t * IDFt \quad (2)$$

d is the d-th document, t is the t-th term of the keyword, tf is the term frequency or word frequency, and W is the weight of the d-th document against the t-term. After each document's weight (W) is known, a sorting process is carried out where the greater the value of W, the greater the degree of similarity of the document to the word you are looking for, and vice versa.

After calculating each document's W weight, calculate the query relevance weighting using (2). From the query relevance values and rankings obtained in Table 1, documents with the highest query relevance weight are displayed sequentially based on their ranking, namely D3, D4, D6, D1, D2, D8, D7, and D5. The query relevance value will be compared with the sentence similarity value for summary extraction.

Table 1. Query relevance value

	D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
Cosine	0.468	0.459	0.678	0.669	0	0.574	0.139	0.150
Rank	#4	#5	#1	#2	#8	#3	#7	#6

The vector space model measures the similarity between a document and a query [33]. In this model, queries and documents are considered vectors in an n-dimensional space, where n is the sum of all the terms in the lexicon. The lexicon is a list of all the terms in the index. One way to overcome this in the vector space model is to expand the vector. The expansion process can be performed on query vectors, document vectors, or both of these vectors. The relationship between words in databases, documents, and keywords [34].

Cosine similarity is used to calculate the query relevance approach to documents. Determining the relevance of a query to a document is seen as a measurement of the similarity between the query vector and the document vector. The greater the similarity value of the query vector with the document vector, the more relevant the query is to the document [35][36].

When the engine receives a query, it will build a vector Q $(wq1, wq2, \dots, wqt)$ based on the terms in the query and a vector D $(di1, di2, \dots, dit)$ of size t for each document. In general, cosine similarity (CS) is calculated using the cosine measure formula [37][38]. This study calculates it using cosine similarity, namely the similarity approach between documents. This study measures the distance between the two documents (di and dj), using the cosine similarity formula to calculate the similarities between documents. In vector space, the document model is represented in the form $d = \{w1, w2, w3, \dots, wn\}$ where d is the document and w is the weight value of each term in the document.

The cosine 0° is one and is less than 1 for every other angle. Thus two vectors with the same orientation have a similarity cosine of 1, and two vectors at 90° have a similarity of 0. Cosine similarity is mainly used in positive space, where the result is bounded by (0,1).

$$Cosine \rightarrow sim\,(dj, q) = \frac{\overrightarrow{dj} \cdot \vec{q}}{|\overrightarrow{dj}| \cdot |\vec{q}|} = \frac{\sum_{=1}^{t}(Wij.Wiq)}{\sqrt{\sum_{=1}^{t} Wij^2 \cdot \sum_{=1}^{t} Wiq^2}} \quad (3)$$

t is a word in the database, d is a document resulting from splitting sentences, and q is a keyword in the abstract.

$$Cosine(Di) = sum(kk2.Di)/[sqrt(kk2) * sqrt(Di2)] \quad (4)$$

Cosine similarity is used to calculate sentence similarity weights, where each document is compared to others. The flow of calculating sentence cosine similarity is the same for calculating the query relevance weights using (4). Table 2 shows the results of the sentence similarity weighting calculation resulting from the cosine similarity calculation. The results obtained on sentence similarity weight values are used to calculate the MMR iteration by comparing the results of query relevance weights and sentence similarity.

Table 2. Sentence similarity weight values

	D1	D2	D3	D4	D5	D6	D7	D8
D1		0.255	0.458	0.375	0	0.207	0.215	0.270
D2	0.355		0,000	0,000	0,000	0,000	0,000	0,000
D3	0.458	0,000		0,000	0,000	0,000	0,000	0,000
D4	0.375	0,000	0,000		0.204	0,000	0.100	0.097
D5	0	0	0	0.204		0.219	0.128	0.107
D6	0.207	0,000	0,000	0,000	0.219		0.162	0.302
D7	0.215	0,000	0,000	0.100	0.128	0.162		0,000
D8	0..270	0	0,000	0.097	0.107	0.302	0,000	

Summary extraction was performed using (5). The MMR calculation is done by comparing the query relevance results and sentence similarity results. Documents have high marginal relevance if

the document is relevant to the contents of the document and has the maximum weight similarity with the query. The final value is given to the S_i sentence in MMR calculated by (1).

$$MMR = argmax\ [\lambda * Sim1\ (SiQ) - (1 - \lambda) * \max Sim2\ (Si, Summ)] \quad (5)$$

Si is a sentence in the document, while $Summ$ is a sentence selected or extracted. The coefficient λ is used to adjust the value combinations to emphasize the sentence's relevance and reduce redundancies. In this study, $Sim1$ and $Sim2$ are two similarity functions that represent the similarity of sentences in all documents and choose each sentence to be used as a summary. $Sim1$ is the Si sentence similarity matrix to the query, while $Sim2$ is the Si sentence similarity matrix to the sentence [31].

The parameter value λ is from 0 to 1 (range [0,1]). When the parameter λ=1, the MMR value obtained will tend to be relevant to the original document. When λ=0, the MMR value obtained tends to be relevant to the previously extracted sentences. Therefore, a linear combination of the two criteria is optimized when the value λ is in the interval [0,1]. For summarizing small documents, such as news, use the parameter value λ = 0.7 or λ = 0.8 because it will produce a good summary [39]. To get relevant summary results, we set the value λ to a value that is closer to λ. The sentence with the highest MMR value will be repeatedly selected into the summary until the desired summary size is reached as in Table 3.

Table 3. MMR Iteration Results

Iteration	D1	D2	D3	D4	D5	D6	D7	D8
1	0.283	0.296	0.451	0.460	-0.044	0.399	0.068	0.060
2	0.135	0.166	0.269	-	-0.079	0.259	0.012	-0.013
3	0.016	0.062	-	-	-0.107	0.147	-0.034	-0.071
4	-0.079	-0.022	-	-	-0.129	-	-0.070	-0.117

Because in the MMR calculation, the values taken as a result of the iteration are more significant than 0, the iteration stops at the 4th iteration because all values are less than 0. Then the values from documents D4, D3, and D6 are considered relevant for summary results.

Table 4. The result of the maximum MMR iteration MMR weight

Iteration	ID	MMR
MMRmax(1)	D4	0.46
MMRmax(2)	D3	0.269
MMRmax(3)	D6	0.147

Table 4 shows the maximum iteration MMR weight obtained from the iteration calculation. Iterations are carried out as many times as the number of documents resulting from sentence splitting, but the one with a positive value or 0 to 1 is taken as a summary. The MMR calculation results show that the document is a summary based on the sequence the highest sentence MMR weight is in Table 5.

From the results of the maximum MMR iteration MMR, it has been determined the order of the relevant documents to be used as a summary, and these documents are sorted by highest to lowest value between weights 0 to 1. Moreover, higher results will place the initial position in the summary.

Because of the results of the maximum marginal relevance calculation, the highest value is taken from all iterations. Documents (D4), (D3), and (D6) are the most relevant and are considered sentences that match the keywords or queries between documents.

The maximum marginal relevance for summary extraction can be seen in Algorithm 1. Lines 1 to 6 of Algorithm 1, delete and create tables starting from the cosine, nmr, and summary tables. The next stage is to call data in the cosine table as in program line 7. Then proceed with reading records in the form of repetition as much as the number of data in the eighth program line.

Lines 9 to 15 determine the number of documents stored in the cosine table with a call value field based on the value of the paper. Furthermore, in program line 16, it is repeated for the total number of documents. In this iteration, we call the cosine table with the SQL command, which is in the 17th program line. Lines 18 to 24 are calculated; the results of the MMR calculation are stored in the MMR table, which is located in line 25. In addition, data updates in the table are also performed.

Table 5. Summary extraction

ID	Document
D4	In the Student Affairs Section of SMA Negeri 1 Tarakan, the process of class promotion and student majors is still carried out in a simple manner by holding meetings and data is processed and stored using Microsoft Excel, so it takes a long time to calculate the process of class promotion and student majors due to the large number of students that must be handled by SMA Negeri 1 Tarakan, so the quality of the results of the process of increasing majors is less accurate, slow and tends to experience differences in decisions between students.
D3	The Student Affairs Section is also a center for processing student data and is also tasked with determining the process of grade promotion and student majors at SMA Negeri 1 Tarakan.
D6	The application in this program starts from the decision tree process for class increases, the decision tree process for the Science majors (Natural Science), Social Sciences Majors, and Language Majors, Student Entry Processes, Class Promotion Processes, Major Processes, and Reports Class Promotion and Majors Report.

Lines 26 to 31 are called the MMR table by reading the document field and storing it in the summary table. Then delete the cosine table based on the document field, delete the mmr table, and create the MMR table as in the 32nd to 34th program line. Lines 35 to 37 call and read summary tables accommodated in the variable data_mmr in the form arrays. The program line 38 is a sentence variable with an empty value to combine the following sentence. Lines 39 to 45 are summary table readings with SQL commands based on the final field of more than 0, which is repeated as much as the sum of the summary data according to the results of the SQL search. Then the sentence is combined with the previous sentence repeatedly.

The summary evaluation is measured by comparing the manual and automated summaries [41]. Manual summaries were obtained from manual summaries of 20 respondents and calculated with precision, recall, and f-measure values as in (6) to (8).

$$Precision = (\#summarization\ sentence \cap manual\ summary) \ / \ \Sigma summarization\ sentence \quad (6)$$

$$Recall = (\#summarization\ sentence \cap manual\ summary) \ / \ \Sigma manual\ summary\ sentence \quad (7)$$

$$f-measure = (2 * Precision * Recall) \ / \ (Recall + Precision) \quad (8)$$

Algorithm 1: Maximum marginal relevance for summary extraction

```
alter file cosine drop field value
alter file cosine and add field value as real
drop file mmr
create file MMR has field document, last
drop file summary
create a file summary that has a field document, last
READ all fields and count as field name total (from file cosine)
WHILE(row is not empty)
    total ← GET field total
END WHILE
FOR no ←1 UNTIL no<=total AS no=no+1
   READ document, value where field value in \
       READ maximum value of field value (from file cosineperkal) \ where
   field document has a string like “no”
   WHILE (row is not empty)
       value ← GET field value
       document ← "D".no
   ENDWHILE
   WRITE file cosine SET field valuekal=value where field \ document=document
   FOR no1 ←1 UNTIL no1<=total AS no1=no+1
       READ all fields (from file cosine)
       WHILE(row is not emmpty)
           document ← GET field document
           value ← GET field value
           valuekal ← GET field valuekal
           left ← 0.8 * value
           right ← ((1-0.8)*valuekal)
           finalResult ← left-right
       ENDWHILE
       WRITE file mmr SET field document= document, field last=finalResult
       WRITE file cosine SET field value= finalResult where field \
   document=document
       READ document, last where field last in \
       READ maximum value of field last (from file mmr)
       WHILE (row is not empty)
           document ← GET field document
           last ← GET field last
       ENDWHILE
       WRITE file summary SET field document= document, \ field last=last
       delete file cosine where field document=document
       drop file mmr
       create table mmr has fields document, last
       READ all fields from summary
       WHILE (row is not empty)
           data_mmr[] ← GET field document + " = " + field last
           SET sentence1 ← "";
       END WHILE
       READ all fields where field last > 0 (from file summary)
       WHILE (row is not empty)
           document ← GET field document
           END WHILE
       READ all fields where field code=document (from sentence)
       WHILE(row is not empty)
           sentence ← GET field sentence
           sentence1 ← sentence1 + " " + sentence + "."
       END WHILE
   ENDFOR
ENDFOR
```

III. Result and Experiments

The data used in the experiments consisted of 200 final project and student thesis abstract documents obtained from the STMIK PPKIA Tarakan Library. Testing is done by entering the contents of the student's final project abstract and abstract keywords. The query is a keyword of the abstract. Sentences taken as a summary represent queries and have a maximum MMR [40] weight between the maximum weight values of 1 to a minimum of 0. The more words similar to the query, the greater the chance for data to be retrieved as a summary. Table 6 shows an example of an evaluation calculation using three documents taken randomly from the data abstract document.

Table 6. Automated summarization and manual summarization

Abstract Id	Summarization of our Model	Manual Summarization
K098	2,3,5	2,4,5
K101	3,4	1,3
K104	11,12,13	12,13

From the summarization results, a comparison was made with the respondents' manual summary. The recall, precision, and f-measure can be seen in Table 7. Table 7 shows the calculation results obtained from precision, recall, and f-measure calculations. The average obtained from these calculations produces an average for precision of 61%, recall of 72%, and f-measure of 66%.

Table 7. Results of example calculations of precision, recall, f-measure comparison of summarization and manual summary

Abstract Id	Precision	Recall	F-Measure
K098	67%	67%	67%
K101	50%	50%	50%
K104	67%	100%	80%
Average	61%	72%	66%

Table 8 summarizes the results of 200 abstract documents of student final assignments. The summary of this model is then compared with the manual summary, which has been done by 20 people summarizing the manual with a summary of 200 abstract documents.

Table 8. Summarization results

Abstract Id	Number of Documents	Summary (Doc ID)	Abstract Id	Number of Documents	Summary (Document ID)
K001	2	7,8	K101	2	3,4
K002	1	10	K102	1	1
K003	2	1,2	K103	2	1,7
K004	1	1	K104	3	11,12,13
K005	2	1,4	K105	2	5,15
:	:	:	:	:	:
:	:	:	:	:	:
K095	2	1,3	K195	2	1,4
K096	1	3	K196	3	2,3,5
K097	2	1,4	K197	1	1
K098	3	2,3,5	K198	3	1,2,6
K099	2	6,10	K199	2	1,11
K100	2	1,2	K200	3	3,5,8

Table 9. Comparison results

Code	Overlap	Precision	Recall	F-Measure
K001	2	100%	67%	80%
K002	1	100%	50%	67%
K003	1	50%	33%	40%
K004	1	100%	33%	50%
K005	2	100%	50%	67%
:	:	:	:	:
:	:	:	:	:
K195	1	50%	50%	50%
K196	2	67%	100%	80%
K197	1	100%	50%	67%
K198	2	67%	67%	67%
K199	2	100%	67%	80%
K200	2	67%	100%	80%
Average		88%	61%	70%

Table 9 shows that the results of the comparison between summarization and manual summaries have an average recall value of 61%, precision of 88%, and f-measure of 70%. Table 10 shows the comparison between the MMR summary result and another model [41]. As seen in Table 10, the Bert2-GPT-Id has a shorter summary than MMR. However, the MMR summary has more comprehensive results than the baseline. In other words, MMR has better performance than Bert2-GPT-Id.

Table 10. An example of MMR test compared to Bert2-GPT-Id

Code	Abstract	MMR	Bert2-GPT-Id
	Modeling is a real system representation of objects by taking a mathematical form and a logical relation. In general, a simulation is defined as a dynamic representation of a portion of the real world using a computer and running at a certain time. One of the modeling techniques is Discrete Event Simulation (DES), modeling a system that changes every unit of time. This method is stochastic, dynamic, and discrete-event. Many fast food restaurants offer a variety of menus and services to satisfy consumers. Kentucky Fried Chicken Restaurant, Tarakan Branch, is one of the most popular fast food restaurants. The increasing number of users of delivery services and different distances, of course, the travel time is also different, resulting in the emergence of new problems in the delivery process. The problem that often occurs at the Tarakan Branch KFC Restaurant is that at certain times KFC receives orders from very many consumers and can make the process of sending orders to consumers slow due to limited employees who specifically handle message delivery services. This will create a queue in the process of sending orders. In this final project, a discrete event simulation model will be implemented using a combination of Fixed-Increment Time Advance and Next-Event Time Advance to overcome problems that occur at the Kentucky Fried Chicken restaurant, Tarakan Branch, using the Delphi 7.0 programming language.	In this final project, a discrete event simulation model will be implemented using a combination of Fixed-Increment Time Advance and Next-Event Time Advance to overcome problems that occur at the Kentucky Fried Chicken restaurant, Tarakan Branch, using the Delphi 7.0 programming language.	The simulation is based on the subject of a real system of objects using a computer and runs at a certain time.
Keywords	discrete events simulation, message delivery, kentucky fried chicken		

IV. Conclusion

Several conclusions are obtained from the discussion and experiments previously conducted in this study. Documents with the highest maximum marginal relevance value from the calculation will be taken as a summary. Sentences taken as a summary represent similar sentences in documents with queries and similarities between sentences in documents. The maximum marginal calculation is done by calculating iterations between combinations of query relevance and sentence similarity matrices. Calculation of query relevance weights is the weight of comparing similarity between queries to documents, while sentence similarity is the weight of the results of comparison of similarities between documents. Vector space modeling is used to query relevance and cosine similarity for sentence similarity.

From the results of the lambda test with a comparison between the lambda values of 0.8, lambda 0.3, and lambda 0.9, it can be concluded that using a lambda value closer to 1 produces a more relevant summary. The results of the experiments are an average precision of 88%, recall of 61%, and f-measure of 70% based on a comparison between the summarization and the manual summary. The test data was taken from 200 student final assignments and thesis documents. Furthermore, used as data in the summarization and manual summary, the summarization results data are compared with the manual summary to obtain accurate results. Moreover, the time needed to summarize one document depends on the number of sentences obtained from document splitting. The more sentences in the document, the longer it takes to summarize.

Some future works are as the results of the comparison with the manual summary show that several abstracts have a low f-measure value because the query sometimes does not describe the content. The retrieved sentences are not in good sentence order. Also, it is recommended to use a generator for abstract keywords. Quality measurement with other parameters, such as F-Score and NMI, is also possible in future research.

Declarations

Author contribution

All authors contributed equally as the main contributor of this paper. All authors read and approved the final paper.

Funding statement

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest

The authors declare no known conflict of financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Additional information

Reprints and permission information are available at http://journal2.um.ac.id/index.php/keds.

References

[1] E. I. Setiawan, V. Natalie, J. Santoso, and K. Fujisawa, "Sequential pattern mining to support customer relationship management at beauty clinics," *Bulletin of Social Informatics Theory and Application*, vol. 6, no. 2, pp. 168–176, 2022.

[2] M. F. Mridha, A. A. Lima, K. Nur, S. C. Das, M. Hasan, and M. M. Kabir, "A survey of automatic text summarization: Progress, process and challenges," *IEEE Access*, vol. 9, pp. 156043–156070, 2021.

[3] M. Wang, X. Wang, and C. Xu, "An approach to concept-obtained text summarization," in *IEEE International Symposium on Communications and Information Technology, 2005. ISCIT 2005.*, 2005, pp. 1337–1340.

[4] E. S. Negara and D. Triadi, "Topic modeling using latent dirichlet allocation (LDA) on twitter data with Indonesia keyword," *Bulletin of Social Informatics Theory and Application*, vol. 5, no. 2, pp. 124–132, 2021.

[5] E. Hovy, "Text Summarization Chapter 32," *Information Sciences Institute of the University of Southern California*, 2003.

[6] H. Haviluddin and R. Alfred, "Big data: issues, trends, problems, controversies in ASEAN perspective," *Bulletin of Social Informatics Theory and Application*, vol. 3, no. 2, pp. 80–93, 2019.
[7] B. Prasetyo, F. S. Aziz, K. Faqih, W. Primadi, R. Herdianto, and W. Febriantoro, "A review: evolution of big data in developing country," *Bulletin of Social Informatics Theory and Application*, vol. 3, no. 1, pp. 30–37, 2019.
[8] J. K. Lê and T. Schmid, "The practice of innovating research methods," *Organ Res Methods*, vol. 25, no. 2, pp. 308–336, 2022.
[9] 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.
[10] S. Tuhpatussania, E. Utami, and A. D. Hartanto, "Comparison Of Lexrank Algorithm And Maximum Marginal Relevance In Summary Of Indonesian News Text In Online News Portals," *Jurnal Pilar Nusa Mandiri*, vol. 18, no. 2, pp. 187–192, 2022.
[11] J. Goldstein and J. G. Carbonell, "Summarization:(1) using MMR for diversity-based reranking and (2) evaluating summaries," in *TIPSTER TEXT PROGRAM PHASE III: Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, 1998, pp. 181–195.
[12] D. Gunawan, S. H. Harahap, and R. F. Rahmat, "Multi-document summarization by using textrank and maximal marginal relevance for text in Bahasa Indonesia," in *2019 International conference on ICT for smart society (ICISS)*, 2019, pp. 1–5.
[13] D. P. Purbawa, R. N. E. Anggraini, R. Sarno, and others, "Automatic Text Summarization using Maximum Marginal Relevance for Health Ethics Protocol Document in Bahasa," in *2021 13th International Conference on Information & Communication Technology and System (ICTS)*, 2021, pp. 324–329.
[14] Y. Mao, Y. Qu, Y. Xie, X. Ren, and J. Han, "Multi-document summarization with maximal marginal relevance-guided reinforcement learning," *arXiv preprint arXiv:2010.00117*, 2020.
[15] P. Gupta, S. Nigam, and R. Singh, "A Ranking based Language Model for Automatic Extractive Text Summarization," in *2022 First International Conference on Artificial Intelligence Trends and Pattern Recognition (ICAITPR)*, 2022, pp. 1–5.
[16] A. Mahajani, V. Pandya, I. Maria, and D. Sharma, "Ranking-based sentence retrieval for text summarization," in *Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS-2018*, 2019, pp. 465–474.
[17] X. Jiang, X.-Z. Fan, Z.-F. Wang, and K.-L. Jia, "Improving the performance of text categorization using automatic summarization," in *2009 International Conference on Computer Modeling and Simulation*, 2009, pp. 347–351.
[18] S. Cahyawijaya *et al.*, "NusaCrowd: Open Source Initiative for Indonesian NLP Resources," *arXiv preprint arXiv:2212.09648*, 2022.
[19] M. Hassel, "Evaluation of automatic text summarization," *Licentiate Thesis, Stockholm, Sweden*, pp. 1–75, 2004.
[20] E. Hovy and C.-Y. Lin, "Automated text summarization in SUMMARIST, Advances in Automatic Text Summarization." MIT Press, 1999.
[21] M. O. El-Haj and B. H. Hammo, "Evaluation of query-based Arabic text summarization system," in *2008 International Conference on Natural Language Processing and Knowledge Engineering*, 2008, pp. 1–7.
[22] Y. Mao, "Guided text summarization with limited supervision," 2022.
[23] A. P. Widyassari *et al.*, "Review of automatic text summarization techniques & methods," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 4, pp. 1029–1046, 2022.
[24] R. A. García-Hernández and Y. Ledeneva, "Word sequence models for single text summarization," in *2009 Second International Conferences on Advances in Computer-Human Interactions*, 2009, pp. 44–48.
[25] R. M. Losee, "Term dependence: A basis for Luhn and Zipf models," *Journal of the American Society for Information Science and Technology*, vol. 52, no. 12, pp. 1019–1025, 2001.
[26] B. Toth, D. Hakkani-Tür, and S. Yaman, "Summarization-and learning-based approaches to information distillation," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010, pp. 5306–5309.
[27] S. Basak, M. D. D. H. Gazi, and S. M. Mazharul Hoque Chowdhury, "A Review Paper on Comparison of Different Algorithm Used in Text Summarization," *Intelligent Data Communication Technologies and Internet of Things: ICICI 2019*, pp. 114–119, 2020.
[28] D. Yadav *et al.*, "Qualitative analysis of text summarization techniques and its applications in health domain," *Comput Intell Neurosci*, vol. 2022, 2022.
[29] S. Xie, *Automatic extractive summarization on meeting corpus*. The University of Texas at Dallas, 2010.
[30] S. Xie and Y. Liu, "Using corpus and knowledge-based similarity measure in maximum marginal relevance for meeting summarization," in *2008 IEEE international conference on acoustics, speech and signal processing*, 2008, pp. 4985–4988.
[31] N. Yusliani, R. Primartha, and M. D. Marieska, "Multiprocessing Stemming: A Case Study of Indonesian Stemming," *International Journal Computer and Applications (IJCA)*, vol. 182, no. 40, pp. 15–19, 2019.
[32] M. I. Aziz, "Development Program Application To The Measurement Of Documents Resemblance Text mining, TFIDF, And Vector space model Algoritm," *Undergraduate Program, Faculty of Industrial Engineering, Gunadarma University*, 2010.

[33] G. Patil and A. Patil, "Web information extraction and classification using vector space model algorithm," *Int. J. Emerg. Technol. Adv. Eng*, vol. 1, no. 2, 2011.
[34] J. Golstein, "Genre oriented summarization," *Unpublished doctoral thesis submitted to Carnegie Melon University. Received in March*, vol. 2, p. 2019, 2008.
[35] I. R. Musyaffanto, G. B. Herwanto, and M. Riasetiawan, "Automatic Extractive Text Summarization for Indonesian News Articles Using Maximal Marginal Relevance and Non-Negative Matrix Factorization," in *2019 5th International Conference on Science and Technology (ICST)*, 2019, pp. 1–6.
[36] J. D. Kapoor and K. K. Devadkar, "Generating Auto Text Summarization From Document Using Clustering," *Int. J. Appl. Eng. Res. Dev.*, vol. 4, no. 2, pp. 31–34, 2014.
[37] M. Chen and Y. Song, "Summarization of text clustering based vector space model," in *2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design*, 2009, pp. 2362–2365.
[38] R. Singh and S. Singh, "Text similarity measures in news articles by vector space model using NLP," *Journal of The Institution of Engineers (India): Series B*, vol. 102, pp. 329–338, 2021.
[39] T. Xing, Z. Xiangxian, G. Shunli, and Z. Liman, "Automatic summarization of user-generated content in academic Q&A community based on Word2Vec and MMR," *Data Analysis and Knowledge Discovery*, vol. 4, no. 4, pp. 109–118, 2020.
[40] 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.
[41] S. Cahyawijaya *et al.*, "NusaCrowd: Open Source Initiative for Indonesian NLP Resources," *arXiv preprint arXiv:2212.09648*, 2022.