A relevant document search system model using word2vec approaches

Selina A Savitri, A Amalia*, M A Budiman

Department of Computer Science, Universitas Sumatera Utara, Jl. dr. Mansur No. 9-A, Kampus USU, Medan 20155, Indonesia

Email: amalia@usu.ac.id

Abstract. USU Repository is an institutional digital information system provided by Universitas Sumatera Utara (USU) that preserves and distributes academic papers, such as thesis and dissertation, from all departments in USU. A search box is provided to help search relevant topics from this repository. However, sometimes the search results returned were irrelevant and did not satisfy the user's expectations. One of the causes for this situation is that the search engine could not perform optimally, particularly if the query were long and complicated. One approach that can be used to solve the problem is using semantic search. Semantic search is an information retrieval process from a sentence that involves understanding the results returned by natural language processing. In light of this approach, this paper aimed to propose a semantic search to seek the relevance between the user's input query and academic papers returned as search results. This study implemented word2vec method in converting sentences into vectors. This study indicated average precision scores for small datasets as 46% and 73% for larger datasets.

1. Introduction

Along with the internet's rapid growth, the number of stored digital text documents has increased rapidly and reached an astonishing level. USU repository as a digital information system provided by USU serves USU academics in collecting, storing, and sharing academics papers published by USU. To assist users in finding relevant papers, the system provided a search mechanism based on keywords typed in by the user in the search box. The search results would be optimal if the input query matched the available keywords from the paper. However, sometimes the search results did not match the user's expectations or perceptions. One of the causes for this situation was the searching algorithm could not provide an optimal result if the query was long and complicated. The longer a query is, the less relevant it is [1]. Classic retrieval models such as TF-IDF, which use a bag-of-words approach, could not capture contextual information from a document or query [2]. This happened because the search process only matched words in a document with keywords in the query and could not handle synonym search.

One approach that can be used to solve the problems is to use the semantic search method. Semantic search can understand the meaning of a word and understand the logical relationship between words. This method may produce better information satisfying the user's wishes, including the synonym [2]. By recognizing the relationship between words in query and words in the dataset, the search results may produce better results [3].

Word embedding, also known as distributed word representation, is a method to represent words into a high-dimensional vector [2]. Word embedding can capture the semantic and syntactic similarities...
between the words. Although word embedding produces high-quality word representations, but the implementation in a longer text, such as sentences, paragraphs, or documents, still represents a huge challenge [4]. The most common method to obtain word embedding vectors for longer texts is by calculating the average word vectors of the sentence [5] or implementing addition of the word vectors from the sentence [6].

Word2Vec is a model for obtaining word embedding developed by Mikolov et al. in 2013 [2]. Word2Vec uses two architectures, namely Continuous Bag of Words (CBOW) and Skip-GRAM. Word2Vec does very well in the semantic representation of words. Usually, the performance of word embedding depends on the corpus size. The bigger size of the corpus will generate better performance of word embedding [7]. Based on this assumption, this study proposed a better document search system in the Computer Science domain using the Word2Vec model. Word embedding was trained using a specific dataset obtained by web crawling the USU Repository, particularly in the Computer Science domain.

This paper is organized as follows: Section II provided the related works in a word embedding. Section III defined the research methodology utilized to build a search system implementing the word embedding method. Section IV discussed the results obtained from the research. Finally, section V concluded the whole research with a summary.

2. Related Works
Semantic-based search engines have been built previously with a different type of embeddings [1]. This research proved that when the query was longer, the system-search results' accuracy would decrease. The work proposed a similarity value of two embedding vectors calculation by using a neural network model. This research reached an accuracy level of up to 84%, while the validation loss obtained after applying the neural network model reached 63%.

Other work-related to similar problems with the Word2Vec model has been conducted [8]. The work covered the problem of song recommendation for cross-language songs between the Indonesian and English languages. It proposed a recommendation system with Word2Vec and query expansion with TF-IDF using a relatively small dataset. Based on the test results, the system's average precision value is 0.38, with the best window size parameter of 3. Research for sentence similarity focused on the semantic of sentences in the Mandarin language for finance texts [10]. The researcher applied word embedding by calculating cosine similarity scores for sentence pairs that had semantic similarities. The results showed that Word2Vec is better than GloVe, with optimal parameters for Word2Vec using CBOW, window size 6, and dimension 400 in a medium-sized corpus (10 million character tokens).

Research conducted by Bintana et al. in 2018 [2] focused on problems in the question-and-answer community where the system must find the same question semantically between new questions (queries) questions in the archive. The researcher applied the CNN word embedding model for semantic modeling of question sentences. The result of the mean average precision (MAP) calculation from the CNN model was 0.422.

3. Methods
In this section, we explained the process from dataset collection until the assessment process.

3.1. Dataset Collection
The first step of processing is collecting a dataset. The dataset is a set of documents obtained utilizing web crawling and scraped from USU Repository (repositori.usu.ac.id/), particularly in the computer science topic. This study only fetched the title and abstract of the academic papers published in the USU Repository.

There were 664 documents successfully crawled from the 774 theses in the repository. The number of tokens in the crawling stage was 122,856. Before continuing to later stages, the crawling results were checked manually. Theses that only had a title were deleted from the dataset.
3.2. Word Embedding
The next step was word embedding training using the Word2Vec model. The training result was pre-trained word vectors that would be used at a later stage.

3.2.1. Word2Vec. In this research, the authors used the Continuous Bag of Words (CBOW) model. Hyperparameters that were used for training in this research are:
1. The parameter "sg" is the algorithm parameter used. In this study, the CBOW algorithm was used, so the value of sg = 0.
2. The parameter "hs" is a parameter for the optimization function. In this study, the Softmax hierarchy function was used, so the value of hs = 0.
3. The parameter "workers" is set with a multiprocess value based on the number of CPUs, so that several threads can be run in parallel. In this study, the number of threads used was 12.
4. The parameter "size" is a parameter to determine the dimensions of the embedding. This study used a dimension of 300. So in the hidden layer, 300 nodes will be used. Fig 1 shows the illustration of the CBOW.

![Figure 1. Illustration for CBOW with 300 dimensions](image)

5. The parameter "min_count" has a value of 3. It aimed to ignore words that have a frequency of occurrence below 3.
6. The parameter "window size" has a value of 10.

3.3. Relevant Documents Search
After generating the word vectors, each document in the dataset will be processed to get the representation vectors. The query will be input and processed to obtain the query's representation vectors.

3.3.1. Preprocessing Queries.
Generally, a query consisted of a single keyword or more. If a query had more than one word, it could not be represented into vectors because one-word vector only represented one word. Therefore, this research proposed a method of concatenating vectors from each word in a query or document to yield the representation vectors. Fig 2 shows an example of this process.
3.3.2. Documents Searching.

The next step is implementing a relevant documents search method based on semantic similarity between query and documents in the dataset. Semantic similarity is calculated using the cosine similarity method. Cosine similarity measures the similarity between two vectors of an inner product space [9]. Two sentences which have similar meaning will have cosine similarity score close to 1 [1]. Equation 1 is used to calculate the similarity between two vectors.

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}} = \frac{x \cdot y}{||x|| ||y||} \quad (1)$$

where:
- $CosSim(x, y)$ = similarity score between documents
- $x$ = query vector
- $y$ = document vector

3.4. Relevancy Measurement

Finally, ten documents with the highest scores of similarity would be shown as the search result. The result of this process will be assessed by human judgment. This process is required to determine the precision score. Equation 2 is the formula for precision that was used in the research.

$$Precision = \frac{\text{number of relevant documents}}{\text{number of total documents retrieved}} \quad (2)$$

4. Results and Discussion

In this research, we aimed to retrieve relevant documents by representing the semantic meaning of the user’s query using the word embedding approach.

After scraping, the dataset file size is 918 KB and comprises 658 documents that consisted of the title and abstract of each thesis. The number of tokens to be trained in the dataset is 121,403 tokens. The training result is pre-trained word vectors, which have a Word2Vec vocab of 4,170.

For testing, as many as 658 thesis title documents would be tested for 10 input queries. Table 1 shows the queries used for testing. The queries were in Bahasa Indonesia.
Each query will be compared against all documents to get the relevant documents. In system testing, the top 10 thesis documents will be selected from the system search results. With the same query, a search will be performed on the USU Repository. The search results from the USU Repository will select the top 10 documents. Four people assessed the document's relevance by answering whether the search results had the same topic as the one that was queried or not. People referred to are lecturers of the USU Computer Science Undergraduate Study Program who are experts in their respective fields, so the validity of each query's relevance assessment is not in doubt by the authors.

Table 2. Search Result on Repository of USU for "Analisis perbandingan kinerja algoritma dalam kompresi file teks" Query

| No. | Relevant Search Results                                                                 |
|-----|-----------------------------------------------------------------------------------------|
| 1.  | Analisis Perbandingan Kinerja Algoritma Goldbach Codes dengan Variable Length Binary Encoding (VLBE) dalam Kompresi File Teks |
| 2.  | Analisis Perbandingan Kinerja Algoritma Shannon Fano dan Levenstein Code pada Kompresi File Video |
| 3.  | Analisis Perbandingan Kinerja Algoritma Start-Step-Stop Code dan Gopala-Hemachandra Code 2 (GH-2(n)) pada Kompresi File Teks |
| 4.  | Analisis Perbandingan Kinerja Algoritma Start/Stop Code dan Algoritma Goldbach G1 Code pada Kompresi File Teks |
| 5.  | Analisis Perbandingan Kompresi Citra Menggunakan Metode Lempel Ziv Welch (LZW) dan Fraktal |
| 6.  | Analisis Perbandingan Kompresi Citra Menggunakan Algoritma Transformasi Walsh-Hadamard dengan Run Length Encoding(RLE) |

| No. | Irrelevant Search Results                                                                                                                                 |
|-----|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.  | Analisis Perbandingan Kombinasi Metode Alpha-Trimmed Mean Filter dan Geometric Mean Filter untuk Mereduksi Noise pada Citra Digital |
| 2.  | Analisis Perbandingan Metode Bandpass Filter dan Bandreject Filter pada Domain Frekuensi untuk Mereduksi Noise pada Citra Digital |
| 3.  | Analisis Perbandingan Pengamanan Citra Menggunakan Algoritma Vigenere Cipher Klasik dan Modifikasi Kombinasi Algoritma Vigenere Cipher dan Rotasi Bit dengan Pembangkitan Kunci Secara Acak |
| 4.  | Analisis Pola Data Penyakit di Rumah Sakit Menggunakan Association Rule dengan Algoritma FP-Growth |
Based on Table 2, the USU Repository managed to find relevant documents even though the query complexity had increased compared to the previous two queries. Documents that were considered relevant were documents with same or similar keywords to the query, such as documents that had the words "Analisis Perbandingan Kinerja Algoritma" and "kompresi file". Meanwhile, irrelevant documents were documents that were not related to the compression field.

Table 3. Search Result on System for "Analisis perbandingan kinerja algoritma dalam kompresi file text" Query

| No. | Relevant Search Results                                                                 | Cosine Similarity Score |
|-----|----------------------------------------------------------------------------------------|-------------------------|
| 1   | analisis algoritma rabin dan algoritma elias omega code dalam pengamanan dan kompresi file teks | 0.992693177660          |
| 2   | perbandingan algoritma boldi-vignaz3 codes dengan algoritma taboo codes dalam kompresi file teks | 0.995090467121          |
| 3   | perbandingan algoritma yamamoto recursive dan punctured elias code dalam kompresi file teks | 0.994688840061          |
| 4   | analisis algoritma rsa dan algoritma elias omega code dalam keamanan dan kompresi file citra | 0.985591037035          |
| 5   | analisis kinerja algoritma elias omega dan algoritma fixed length binary encoding pada kompresi file teks | 0.98549617639          |
| 6   | analisis perbandingan algoritma boldi vigna z3 code dan algoritma even-rodeh code pada kompresi file teks | 0.983301504611          |
| 7   | analisis perbandingan algoritma even-rodeh code dan algoritma fibonacci code untuk kompresi file teks | 0.988082580597          |
| 8   | analisis perbandingan algoritma ternary comma code tcc dan rice code dalam kompresi file citra digital bitmap | 0.984596439306          |
| 9   | analisis perbandingan kinerja algoritma goldbach codes dengan variable length binary encoding vlbe dalam kompresi file teks | 0.986030841948          |
| 10  | analisis perbandingan kinerja algoritma start-step-stop code dan gopala-hemachandra code 2 gh-2n pada kompresi file teks | 0.983459027974          |

| No. | Irrelevant Search Results                                                                 | Cosine Similarity Score |
|-----|----------------------------------------------------------------------------------------|-------------------------|
| 1   | -                                                                                       |                         |

Based on Table 3, the system indicated a significant improvement compared to Table 2. Documents that were considered relevant were documents with the same or similar keywords to the query, such as documents that had the words "analisis algoritma" and "kompresi file". In this query, it is evident that the cosine similarity value, which is close to 1, has a semantic meaning that is almost the same as the query.

After relevance testing, the precision score would be calculated between the system and the USU repository as a comparison to see whether the system is better than the USU repository or not. The final precision calculation is showed in Table 4.
### Table 4. Precision Score

| Query                                                                 | Precision of USU Repository | Precision of System |
|----------------------------------------------------------------------|----------------------------|---------------------|
| algoritma djikstra                                                  | 0%                         | 0%                  |
| web semantik                                                        | 0%                         | 20%                 |
| Analisis perbandingan kinerja algoritma dalam kompresi file teks    | 60%                        | 100%                |
| penerapan algoritma rsa dan algoritma spritz dalam skema hybrid cryptosystem | 20%                        | 90%                 |
| analisis perbandingan kerja algoritma di kompresi file teks         | 0%                         | 50%                 |
| Implementasi Algoritma Antt Colony pada masalah travelling salesman | 0%                         | 0%                  |
| masalah travelling salesman                                         | 90%                        | 20%                 |
| pengamanan file di android analisa                                 | 30%                        | 90%                 |
| perbandingan kompresi image dgn algoritma transformasi              | 0%                         | 80%                 |
| penyangian data teks metode simetri                                | 20%                        | 10%                 |
| **Average:**                                                         | **22%**                    | **46%**             |

Based on the experiments, there were several improvements using word embedding compared to that of Repository USU. Repository USU failed to retrieve documents if there were any misspelling present in the query, while this was not a major issue for the system in this research. Grammar in queries was not important since the purpose of word embedding was to determine the semantic meaning so that word syntactic may be disregarded. From the test results of 10 queries, the system had an average precision value above 40%. The average precision value increased by 24% compared to that of the Repository USU for the top 10 documents' search results. This increase was considered not satisfying, so the authors tested again with a different dataset for training word embedding as evaluation material.

The previous test's difference was the current test used a word embedding model, which is trained using a different dataset. Evaluations with a larger dataset were obtained from research [11]. The dataset contained 627 theses from undergraduate to Ph.D levels at the Faculty of Computer Science and Information Technology USU. The word embedding was trained with the same hyperparameters. The number of words that were trained for the second test was 56,353,916 words. The result of the second testing was shown in Table 5.
Table 5. Precision Score

| Query                                                                 | Precision of First Testing | Precision of Second Testing |
|-----------------------------------------------------------------------|----------------------------|----------------------------|
| algoritma djikstra                                                    | 0%                         | 60%                        |
| web semantik                                                          | 20%                        | 90%                        |
| Analisis perbandingan kinerja algoritma dalam kompresi file teks     | 100%                       | 100%                       |
| penerapan algoritma rsa dan algoritma spritz dalam skema hybrid cryptosystem | 90%                        | 90%                        |
| analisis perbandingan kerja algoritma di kompresi file teks          | 50%                        | 80%                        |
| Implementasi Algoritma Artt Colony pada masalah traveling salesman   | 0%                         | 70%                        |
| masalah travelling salesman                                           | 20%                        | 60%                        |
| pengamanan file di android analisa                                   | 90%                        | 90%                        |
| perbandingan kompresi image dgn algoritma transformasi               | 80%                        | 10%                        |
| penyandian data teks metode simetri                                  | 10%                        | 80%                        |
| **Average:**                                                          | **46%**                    | **73%**                    |

Based on Table 5, a system that used word embedding trained on a larger dataset displayed better precision scores than the previous word embedding. The query results tested in the second testing have had an increase in the average precision value, and there was no decrease in the average precision score.

5. Conclusion
The conclusions of this work were as follows. First, the Word2Vec model could be used as an alternative solution in building a simple, yet relevant document search system. Second, the more words in the query that were similar to the documents searched, the more relevant the results tended to be. Lastly, in this study, the average precision value was 46%, and the average precision value with a larger dataset was 73%.

References
[1] Patel, M. (2019). TinySearch--Semantics based Search Engine using Bert Embeddings. arXiv preprint arXiv:1908.02451.
[2] Bintana, R. R., Faticah, C., & Purwitasari, D. (2018). Pencarian Question-Answer Menggunakan Convolutional Neural Network Pada Topik Agama Berbahasa Indonesia. Ultimatics: Jurnal Teknik Informatika, 10(1), 57-64.
[3] Unik, M., & Ramli, M. Penerapan Metode Pencarian Semantik Dalam Sistem Informasi Pencarian Dokumen Kerja Praktek Dan Skripsi Berbasis Web.
[4] Perone, C. S., Silveira, R., & Paula, T. S. (2018). Evaluation of sentence embeddings in downstream and linguistic probing tasks. arXiv preprint arXiv:1806.06259.
[5] Arora, S., Liang, Y., & Ma, T. (2016). A simple but tough-to-beat baseline for sentence embeddings.
[6] Le, Q., & Mikolov, T. (2014, January). Distributed representations of sentences and documents. In International conference on machine learning (pp. 1188-1196).
[7] Altszyler, E., Sigman, M., Ribeiro, S., & Slezak, D. F. (2016). Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database. arXiv preprint arXiv:1610.01520.
[8] Aldiansyah, G. W., Adikara, P. P., & Wihandika, R. C. Rekomendasi Lagu Cross Language Berdasarkan Lirik Menggunakan Word2VEC. (2019). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN, 2548, 964X.
[9] Si, S., Zheng, W., Zhou, L., & Zhang, M. (2019, June). Sentence Similarity Computation in Question Answering Robot. In *Journal of Physics: Conference Series* (Vol. 1237, No. 2, p. 022093). IOP Publishing.

[10] Putri, S. K. (2020). *Pre-Trained Word Vector Bahasa Indonesia Generation* untuk Bidang Komputer dan Teknologi Informasi.