Bahasa Indonesia pre-trained word vector generation using word2vec for computer and information technology field

Syarifah K Putri¹, A Amalia¹*, E B Nababan², O S Sitompul²

¹Department of Computer Science, ²Department of Information Technology, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Jl. Universitas No. 9-A, Kampus USU, Medan 20155, Indonesia

Email: amalia@usu.ac.id

Abstract. Words embedding or distributed representations is a popular method for representing words. In this method, the resulting vector value is a set of real values with specific dimensions that are more effective than the Bag of Word (BoW) method. Also, the advantages of distributed representations can produce word vectors that contain semantic and syntactic information, so that word vectors with close meanings will have close word vectors. However, distributed representation requires a huge corpus with a long training time. For this reason, many researchers have created trained word vectors that can be used repeatedly. The problem is that the available trained word vectors are usually general domain word vectors. This study aims to form pre-trained word vectors for specific domains, namely computers and information technology. Researchers used a dataset of student scientific papers from the Universitas Sumatera Utara (USU) repository. Researchers used the word2vec model, where the model has two architectures, namely the Continuous Bag-of-Words (CBOW) and Skip-gram. This research's result is word2vec model with the CBOW method is more effective than the Skip-gram method.

1. Introduction

One of the artificial intelligence domain that has more attention lately is Natural Language Processing (NLP). NLP aims to enable computer systems to understand the human language. A simple example of an NLP application is the intuitive ability of a system to group documents according to their categories. NLP techniques that use a rule-based system are ineffective because they require much time and human resources. One of the popular ways today is machine learning, where the process of grouping text such as classification, clustering is carried out based on sample training data. The initial and the most important machine learning stage is to transform the representation of words into vectors. This stage can be done using the bag of word methods such as one-hot representation. In one hot representation, every word found from the document is converted into a vector with a value of 1 and 0, where the vector length corresponds to the vocabulary size. However, this method cannot capture semantics and syntax and has a sparsity problem because each word's dimension is as many as the number of vocabularies in the corpus. In contrast, another method to transform the text into a vector is a distributed representation method, also known as word embedding. This method produces word vectors using a neural network approach. Word embedding is a method for representing words into a vector with real numbers. The advantage of word embedding is that it can calculate the vector value of words based on their closeness of relevance. Therefore the word embedding becomes the forerunner of NLP tasks such as clustering, classification, etc. The popular method of word embedding is word2vec by Thomas Mikolov [1]. The resulting word vector value is a matrix that carries much information such as semantic information, syntax, and also closeness between words [2]. For words that have a similar
meaning, will have a word vector that is close too. The process of training word vectors with word embedding requires a large dataset and takes a long time. However, the advantage of building the results of training word embedding, known as pre-trained word vectors, can be used repeatedly for other NLP tasks such as classification, clustering, etc. Although most of the pre-trained produced is in English, some research like [3] has made pre-trained for many other languages, including Bahasa Indonesia. The study builds general pre-trained word vectors for these languages using Wikipedia translations for multiple languages. Many opinions in generating word embedding tendency to focus on using a huge corpus. However, the corpus specificity has much more influenced to generate a better word embedding than its size [4]. To automatically detect similar words in a specific field, the most important thing is the appropriate corpus's training embedding than the huge general corpus. In contrast to the general domain, there are challenges forming a word vector for a certain field because each word must have a close relationship. So far, many researchers have created word vectors using the Wikipedia corpus. Our hypothesis, we cannot get an optimum result to catch the semantic and syntactic relation in the computer field if we used the general dataset corpus from Wikipedia. It is because there are not many training datasets from Wikipedia about the computer field. Therefore, in this study, we propose the pre-trained Bahasa Indonesia word vector in the computer field. Our contribution, the result of this study, a set of pre-trained word vectors in the computer field in Bahasa Indonesia, can be a language resource in building further NLP tasks in this domain. For instance, to build a search engine for computer articles based on semantics. Furthermore, the methodology of this study also can be implemented by another field. This paper will be organized as follows: Section II will provide the related works in word embedding among several languages. Section III defines the research methodology to build a word vector using the word embedding method. Section IV discusses the result of the word vector using two architectures. Finally, section V will conclude the whole work.

2. Related Works

Many previous studies used word embedding representation in a specific domain. For instance, the study by [5] introduced a domain semantic similarity measure created by the synergistic union of word2vec. Another previous study generated word embedding from social media like Twitter. This study aims to found a health-related classification method. The result has concluded that the word embedding can represent information for words so that a tweet can be divided into groups of words similar to the size of a cluster of words [6]. Another previous study implemented the word2vec to get the contextual meaning of the token [7]. This study used a dataset sourced from the informal language of Bahasa Indonesia in social media. The system gets an accuracy rate of 79.59% to normalize tokens that are not formally compatible with Bahasa Indonesia. Another previous study used an extended of word2vec, which is Doc2Vec, in small corpora. The study aims to compare TFIDF-LSA and Doc2Vec algorithm to classify the document's article in small corpora [8]. Researchers using a small dataset for the study. This study's result is for the cluster performance, which is based on extrinsic measurement; the TDIDF-LSA gets better performance than the Doc2Vec model. Other studies related to word embedding have been conducted for many languages [3]. Researchers construct word vectors for 100 languages. In his research, the researcher took a sample of 9 languages to measure word closeness's accuracy, and it was found that the largest decrease in value was -2.68%. Furthermore, other studies use hotel reviews in Traveloka to search the best hotel using word2vec, glove, and doc2vec [9]. The results of this study that using word2vec Skip-Gram has an accuracy of 91.81%. Previous study was successfully developed a corpus for Indonesian academic language [10].

3. Methods

This research aims to create Indonesian pre-trained word vectors for computers and the information technology field. There are four stages.
Based on figure 1, the first stage in this research is to collect a dataset from the USU repository website using the web crawling method. The dataset is crawled from scientific papers by Computer Science and Information Technology Faculty students uploaded from 2018 - 2020. The number of papers used is 627. The second stage is the pre-processing. There are several steps involved in the pre-processing process:

a. Extract PDF
The raw dataset obtained from USU's repository is in PDF format. The contents of this PDF contain the string, images, and tables. In this study, we extracted string and table contents and removed the images.

b. Merge files
In this stage, the extracted PDF files are transformed into txt format. Further, these files are combined into one large text file that holds all dataset files. This study using the pdfminer library to concatenate all these files into one large file.

c. Data cleaning
Data cleaning is a process of removing unwanted symbols like excess numbers, spaces or whitespace and hyphens such as ",", "(", ")", "[", "]", "{", "}", ".". We also remove the listing and source code program from the dataset. We found many excerpts of the source code program because the dataset is from computer science academic papers. We use regex to clean unwanted symbols. However, we still keep the hyphens or a dash "-" with reason, some words in Bahasa Indonesia using the hyphen as part of the word.

d. Case folding
Case folding is a technique for converting the text in a document into a standard form, for example changing text letters to uppercase or lowercase letters. In this study, all text in the dataset is converted into lowercase form.

e. Sentence tokenize
Tokenization or tokenize is breaking a sentence into tokens. Datasets that have gone through the data cleaning process will be broken down into tokens or words. Sentence tokenization aims to limit word-by-sentence retrieval in the word embedding training process to be included in the input layer.

f. Split word
The split word is to separate a sentence into words. In this study, the researcher used the split library to separate words in a sentence that had been tokenized. The number of words after being tokenized was 7,166,098.

The third stage is training. Training is the core of the process carried out in machine learning. In this study, the training process was carried out for two architectural models: Continuous Bag-of-Words (CBOW) and Skip-gram. The following is an illustration of the CBOW and skip-gram neural networks used in the study.
The first training process is Continuous Bag-of-Words (CBOW) training with the Hierarchy Softmax optimization function. The second training process is Skip-gram training with the Negative Sampling optimization function. Skip-gram predicts the context of a word based on the input given [6]. Meanwhile, CBOW aims to predict one target word given a one-word context, such as the bigram model [7].

The goal is to use two word2vec architectural models to compare which models produce accurate similarities between words. In this study, we implemented word2vec from the gensim library (https://radimrehurek.com/gensim/models/word2vec.html). Some hyper-parameters need to be initialized like window size, dimension, mint_count, SG, HS, workers, and seed. In this study, we used parameter values commonly used by previous researchers to consider that they have done many trials to find the best value. The explanation of the parameters are used for this study as follow:

a. Window size is the maximum distance between the target word and the context words in a sentence. In this study, we used the parameter of windows size 8, 9, and 10.

b. Dimension is the node size of the hidden layer, in this study used 300 dimensions.

c. Min_count is the minimal frequencies of word occurrence in the corpus. In this study, we use min_count 3.

d. Sg is the parameter to choose the word2vec algorithm between skip-gram and CBOW. Value 1 for skip-gram and 0 for CBOW. In this study, we experiment using both algorithms.

e. Hs is the parameter to choose a softmax model between hierarchical softmax and negative sampling. The value 1 to activate the hierarchical softmax meanwhile value 0 to activate the negative sampling model. In this study, we implemented both models.
f. Negative is a parameter to initialize if we choose Hs = 0. The number of negative sampling should be drawn. In this study, we implement negative 5 because it is standard from the library.

g. Workers is a parameter to determine the number of threads used. In this study, we used workers 8 because the total number of CPUs researcher is 8

h. Seed is the initial value for weights randomly. In this study, we used seed = 1

The fourth stage is the evaluation to measure how good the word vector obtained in this study. We implemented the cosine similarity measurement between word vectors and found the nearest words for each word. To evaluate the result, we gave questionnaires to several participants to response to the result.

4. Results and Discussion

This study used two-word embedding architectures, namely Continuous Bag-of-Word and Skip-gram, for model training. The generator library has provided both of these architectures. The number of words used for the test was 7,166,098 words. The following is a 2-dimensional word vector graphic.

![Word Vector in 2 Dimensional Shapes](image)

Fig 4. Word Vector in 2 Dimensional Shapes

a. CBOW

The study used Continuous Bag-of-Word architecture in the model using the hierarchical softmax algorithm. The parameters used are the dimensions of 300, and the windows size used is 8,9,10. When building vocab for dimension 300, the memory used to build vocab is 212,8027 MB, and the length of time used for vocab training is 76.7 s with an epoch of 5.

| Input     | Windows size 8 | Windows size 9 | Windows size 10 |
|-----------|----------------|----------------|-----------------|
| Normalisasi | Normalisasi, menormalisasikan, penyaringan, cleaning, normalisasi, minimum, mikroteks, z-score, filtering | Menormalisasikan, normalisasi, koversi, cleaning, pembobotan, filtering, penyaringan, normalisasi, perhitungan | Normalisasi, menormalisasikan, koversi, mbkghr, penyaringan, pembagian, pengecekan |

Table 1. Testing the CBOW Method and the Hierarchy Softmax of 300 dimensions
The researcher uses the Skip-Gram architecture in the model using the Negative Sampling algorithm. The parameters used are the dimensions of 300, and the windows size used is 8, 9, 10 with epoch 5.

Table 2. Testing the Skip-gram Method and the Negative Sampling of 300 dimensions

| Input      | Windows size 8 | Windows size 9 | Windows size 10 |
|------------|-----------------|-----------------|-----------------|
| Informasi  | Informasi,      | Informasi,      | Informasi,      |
|            | informasi-      | informasi-      | informasi-      |
|            | informasi,      | informasi,      | informasi,      |
|            | gambaran,       | gambaran,       | gambaran,       |
|            | pemahaman,      | pemahaman,      | pemahaman,      |
|            | data,           | data,           | data,           |
|            | infromasi,      | penjelasan,     | penjelasan,     |
|            | pengetahuan,    | pemahaman,      | pemahaman,      |
|            | biometrik       | data,           | data,           |

b. Skip-Gram

The following are the results of testing the word2vec model using words related to computers and information technology. Table 3 and Table 4 explain that the output is related to word input but has differences because each table using a different method.

Table 3. Test Results of the Skip-Gram and Negative Sampling Method

| Input      | Windows size 8 | Windows size 9 | Windows size 10 |
|------------|-----------------|-----------------|-----------------|
| Data       | Data, subsets,  | Data, subsets,  | Data, tetangg,  |
|            | data-data, set.| set, data-data, | label-labelinya,|
|            | iris, validasi | latih-,         | coba., iris,   |
|            | latih, wine,    | mengujinya,     | latih-,        |
|            | tetangg,        | coba., tetangga | training, kdd. |
|            | berlabel        | kelasnya,       | subsets         |
|            |                 | training        |                 |
| Dekompresi       | Kompresi, dekompresi, mendekompresi, kcmpresi, dekompresi-dekripsi, lzw | Kompresi, dekompresi, kompresi, dekompresi-dekripsi, mendekompresi, dekripsi-dekompresi | Kompresi, dekompresi, mendekompresi, didekompresi, lzw, kompresi |
|------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| Kriptografi      | Asimetr, simetri, simetrik, kriptografi, kunci-simetri, kunci-asimetri   | Asimetr, kriptografi, asimetr, simetri, kunci-asimetri                                        | Asimetr, kriptografi, simetri, kunci-asimetri, asimetris          |

Table 4. Test Results of the CBOW and Softmax Hierarchy Method

| Input          | Windows size 8 | Windows size 9 | Windows size 10 |
|----------------|----------------|----------------|-----------------|
| Dekompresi     | Kompresi, dekompresi, kompresi, enkripsi, dekripsi, fraktal          | Kompresi, dekompresi, kompresi, enkripsi, dekripsi, enkripsi.                               | Kompresi, dekompresi, kompresi, enkripsi, dekripsi, lzw         |
| Kriptografi    | Kriptografi, asimetr, simetri, penyandian, simetri, asimetri          | Kriptografi, penyandian, asimetr, simetri, simetri, kriptografi                           | Kriptografi, simetri, asimetr, penyandian, simetri, asimetri    |

The test results in table 3, and table 4 are analyzed by human judgment or participants through several questionnaire form questions. The participants are 7 lectures and students from the Faculty of Computer Science dan Information Technology USU. According to these seven people, the prediction results using the Continuous Bag-of-Words method and the Softmax Hierarchy and using the dimensions of 300 and windows size ten is more accurate than using Skip-gram and Negative Sampling.

The accuracy of the model can also be tested using the cosine similarity method between two words. If the word tested is the same, it will have a value of 1. The cosine similarity distance can be -1 to 1, and sometimes it is 0 to 1, depending on the results of model calculations. The following are system testing results using cosine similarity with the CBOW method and windows size 10.
Table 5. Cosine Similarity CBOW Method and Softmax Hierarchy

| First Parameter | Second Parameter | Vector Results |
|-----------------|------------------|----------------|
| Kriptografi     | Citra            | 0.200529695598712 |
| Kompleksitas    | Kompleksitas     | 0.999999999999999 |
| Bobot           | Training         | 0.291256417626771 |
| Kunci           | Enkripsi         | 0.685052538964685 |
| Geografis       | Matriks          | 0.168788312355349 |

Based on the test results table above, the words "kunci" and "enkripsi" have a cosine similarity value of 0.6805, which means they have a high correlation relationship. In contrast, for the words "geografis" and "matriks" have a cosine similarity value of 0.168, meaning they have a low correlation relationship. To evaluate whether the pre-trained results are correct, the researcher also compares the research word vectors' results with the results of the vector words of Wikipedia in Indonesian and KBBI-Kamus Besar Bahasa Indonesia. Pre-trained Wikipedia has 293,643,490 words and 2,234,150 distinct words. The following is a comparison table between pre-trained researchers and pre-trained Wikipedia with 300 and Windows 8 and using the Continuous Bag-of-Words (CBOW) algorithm and the Hierarchy Softmax.

Table 6. Comparison of Pre-trained Researchers and Wikipedia Pre-trained

| Input          | Researchers Pre-trained | Wikipedia Pre-trained |
|----------------|-------------------------|-----------------------|
| Dekompresi     | Kompresi, dekompresi, kompresi, enkripsi, dekripsi, fraktal | Malfungsi, malafungsi, sumbatan, turbulensi, dehidrasi, penyumbatan |
| Kriptografi    | Kriptografi, asimetris, simetri, penyandian, simetris, asimetri | Enkripsi, algoritme, kriptosistem, algoritma, cipher, otentikasi |

The following is a visualization of word embedding categories based on the input words.

Fig 5. Word Embedding Visualization
Based on the observation, the word vector’s performance depends on the quality of the data corpus. In this study, the performance decreases because the corpus dataset still contains pseudocode and bibliography in its documents. Many papers have pseudocode in it regarding the corpus source from student papers on the computer field. As well as a bibliography in the papers also decreasing the performance of the pre-trained word vector. Pseudocodes The dataset used must not contain program code because the prediction results will be inaccurate. The dataset used still has noise such as pseudocode and bibliography, affecting the quality of results. The period plays an essential role in the dataset because every paragraph in the dataset must be split marked with the period.

5. Conclusion
This study aims to generate a pre-trained Bahasa Indonesia word vector for the computer and information technology field using the word2vec model. The corpus originated from 627 scientific papers in the computer field with numbers of words 7,166,098. Based on the analysis and results of system evaluation, it is concluded that the pre-trained word vector from a specific corpus has a better performance in the field domain compared to the general pre-trained vector like Wikipedia. The study compared the quality of word vector using general pre-trained from Wikipedia using 293,643,490 words. After doing several experiments, the number of dimensions is very influential when predicting words. The dimensions of 300 and windows size ten can produce good word predictions for both architectures, namely CBOW and Skip-gram. The quality of a pre-trained word vector also depends on the corpus’ quality; therefore, the corpus needs to be prepared, such as removing unwanted symbols. In this study, the corpus still has noise, such as pseudocode and bibliography.

Acknowledgments
We gratefully acknowledge that this research is funded by Lembaga Penelitian Universitas Sumatera Utara. The support is under the research grant TALENTA USU of Year 2020.

References
[1] Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning poly based natural language processing. IEEE Computational intelligence magazine, 13(3), 55-75.
[2] Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot: Distributed word representations for multilingual nlp. arXiv preprint arXiv:1307.1662.
[3] Dai, X., Bikdash, M., & Meyer, B. (2017, March). From social media to public health surveillance: Word embedding based clustering method for twitter classification. In SoutheastCon 2017 (pp. 1-7). IEEE.
[4] Kusumawardani, R. P., Priansya, S., & Atletiko, F. J. (2018). Context-sensitive normalization of social media text in bahasa Indonesia based on neural word embeddings. Procedia computer science, 144, 105-117.
[5] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
[6] Vasilev, Ivan., Slater, Daniel., Spacagna, Gianmario., et al. 2019. Python Deep Learning Second Edition. Packt Publishing.
[7] Rong, X. (2014). word2vec parameter learning explained. arXiv preprint arXiv:1411.2738.
[8] A. Amalia, O. Salim Sitompul, E. Budhiarti Nababan, M. Silvi Lydia, and T. Mantoro, “A COMPARISON STUDY OF DOCUMENT CLUSTERING USING DOC2VEC VERSUS TFIDF COMBINED WITH LSA FOR SMALL CORPORAS” J. Theor. Appl. Inf. Technol., vol. 98, p. 17, 2020.
[9] Imaduddin, H., & Fauziati, S. (2019, March). Word Embedding Comparison for Indonesian Language Sentiment Analysis. In 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIIT) (pp. 426-430). IEEE
[10] Kwary, D.A. (2019). A corpus platform of Indonesian academic language. SoftwareX, 102. Elsevier