Automatic Extraction of Indonesian Stopwords

Harry Tursulistyono Yani Achsan[1], Heru Suhartanto[2], Wahyu Catur Wibowo[3], Deshinta A. Dewi[4], Khairul Ismed[5]
Faculty of Computer Science, University of Indonesia, Depok, Indonesia[1, 2, 3]
Faculty of Science & Engineering, Universitas Paramadina, Jakarta, Indonesia[1]
INTI International University & Colleges, Nilai, Malaysia[4]
National Research and Innovation Agency of Republic of Indonesia, Indonesia[5]

***Abstract*—The rapid growth of the Indonesian language content on the Internet has drawn researchers' attention. By using natural language processing, they can extract high-value information from such content and documents. However, processing large and numerous documents is very time-consuming and computationally expensive. Reducing these computational costs requires attribute reduction by removing some common words or stopwords. This research aims to extract stopwords automatically from a large corpus, about seven million words, in the Indonesian language downloaded from the web. The problem is that Indonesian is a low-resource language, making it challenging to develop an automatic stopword extractor. The method used is Term Frequency – Inverse Document Frequency (TF-IDF) and presents a methodology for ranking stopwords using TFs and IDFs, which is applicable to even a small corpus (as low as one document). It is an automatic method that can be applied to many different languages with no prior linguistic knowledge required. There are two novelties or contributions in this method: it can show all words found in all documents, and it has an automatic cut-off technique for selecting the top rank of stopwords candidates in the Indonesian language, overcoming one of the most challenging aspects of stopwords extraction.**

Keywords*—Stopwords extraction; attributes reduction; TF-IDF; large corpus; Indonesian stopwords; NLP

I. Introduction

Stopwords are any common words that carry low information content [1]. Despite their high occurrence, they only add a little semantic data to the document [2]. They are also referred to as negative dictionary or noise words. They cause a small retrieval degree and prediction outcomes. Since they make up a considerable portion of the documents, text-mining tasks will be very computationally intensive. This high computational cost is caused by the dimensionality curse and requires larger computer memory and computational time. Furthermore, in information retrieval experiments, it has been shown that removing stopwords improves precision significantly when compared with when they are not removed [3, 4]. Stopwords also play a significant role in feature extraction [5, 6], topic modeling [7], classification [8], ontology construction [9], and keyword extraction [10].

There are two categories of stopwords: domain-specific and general. Domain-specific stopwords are a set of words that make no discriminant value inside a specific context or domain. They differ from one domain to another domain. For example, the word "learning" could be a stopword in the education domain, or the word "machine" could be a stopword in the machinery domain, but neither of those words is a stopword in the computer science domain. On the other hand, general stopwords are a list of stopwords or stoplists that are not specific to one domain and are usually available to download as a public domain object.

General stopwords are the most used in Natural Language Processing (NLP) because of their availability, and it takes a considerable effort to develop a domain-specific stoplist. It is easier to create a domain-specific stoplist based on a general stoplist by adding and/or removing some terms. General stoplists, however, need to be updated frequently. In addition, over time, the use of some ordinary words has altered subjects on social aspects such as industrial revolution changes, new media, cultural shifts, and education. For these reasons, reviewing, updating, and adjusting existing stoplists is essential [5]. Updating a general stoplist can be done manually, but it takes time and may omit the latest stopwords. This problem can be solved automatically by developing a general stoplist.

Researchers have developed many methods for automatically creating stoplists, especially in English, since decades ago. Since then, many methods have been developed to create English stoplists. In contrast, there are relatively few studies to develop a stoplist for non-English languages like Indonesian. The problem is that Indonesian is a low-resource language, making it challenging to develop an automatic stopword extractor.

There were only two research documents about general stopwords extraction in Indonesian [11, 12]. Those documents show 394 and 330 general stopwords extracted from Kompas daily newspaper. Both of stopwords lists extracted using Term Frequency (TF) method, it is a rare method to use in extracting stopwords. Most researchers use a combination of Term Frequency and Inverse Document Frequency (TF-IDF) in NLP. Unfortunately, TF cannot detect words that occur in all documents and cannot implement threshold to limit the number of generated stopwords.

This research paper aims to solve the problem above and develop an up-to-date general stoplist in the Indonesian language. The method used is crawling recent news from an Indonesian online newspaper's website to gather data and make the required dataset. The stopwords extraction method uses TF-IDF.

This document is organized in this way; the following section comprises a brief coverage of the present literature in the areas of automatic stopwords extraction, the methods used for stopwords extraction, and the experiments. Then we describe the results of this research. We conclude this research document by presenting the advancement of our methods compared to previous works.

II. Related Works

Many methods have been used to develop stoplists. Some of them are frequency-based approach [13], Bidirectional Long Short term memory (BiLSTM) [14], Word Embedding [15], Finite Automata [16], and utilizing characteristic and discriminant analysis [17]. The dataset or corpus used to extract or identify stopwords vary. Some researchers used corpus from an online newspaper [18], social network [19], or patent [20]. As the purpose of this research is to develop an Indonesian general stoplist, only relevant research papers are discussed.

There are three research papers discussing the development of the Indonesian stoplist. One of them only involves developing a cuisine-specific stoplist for the Indonesian language [12]. However, since this research aims to develop a general Indonesian stoplist, only the other two papers are reviewed further here based on their proposed method.

Fadillah Z Tala, in his master thesis [11], proposed an Indonesian stoplist because there was no Indonesian stoplist that could be used in his experiment in information retrieval. In his work, he created the dataset based on articles from the “Kompas daily” newspaper. He downloaded the articles every day for one year long, starting from the beginning of January 2001 until the end of December 2001. The total number of articles was 3160. The result of his experiment was 394 stopwords in Bahasa Indonesia.

Yudi Wibisono has created a stoplist in his coursework [12]. As the source of his dataset, he also used articles from the “Kompas daily” newspaper. He used several hundred articles to create 330 stopwords. The method used was Term Frequency, like Tala’s work, but he removed some words manually.

Tala and Wibisono used the Term Frequency (TF) method to extract stopwords in their work. These days, Term Frequency-Inverse Document Frequency (TF-IDF) is another method that is generally used in information retrieval systems. TF-IDF is one of the traditional methods based on statistics [21]. It has been used in many different applications, such as document clustering [22], text classification [23], detection of domain name generation algorithms [24], and comparing research trends [25]. Term frequency or word frequency is a rarer method used in information retrieval systems compared to TF-IDF.

III. Methods

Different methods were used in each stage of this research. As shown in Fig. 1, the steps for this study were differentiated into three stages: data gathering, pre-processing or data cleaning, and stopwords extraction.

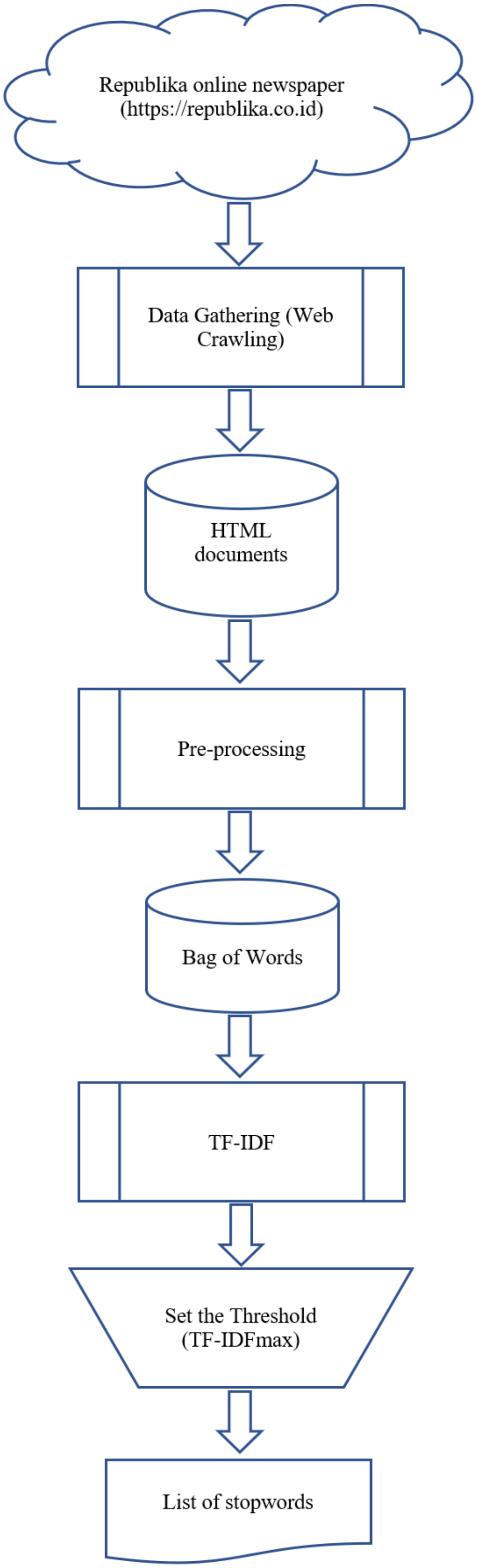
Republika online newspaper
(https://republika.co.id)
Data Gathering (Web Crawling)
HTML documents
Pre-processing
Bag of Words
TF-IDF
Set the Threshold (TF-IDFmax)
List of stopwords

Fig. 1. Steps for this study

This work is different from the previous studies in some stages. First, the data source in the data gathering stage of this research is crawled from the “Republika daily” newspaper, whereas the previous studies used data from the “Kompas daily” newspaper. Moreover, in their studies, they used Term Frequently (TF), but in this work, we used the TF-IDF method, a combination of the Term Frequently and Inverse Document Frequency methods.

A. Data Gathering

The dataset or corpus for this research was gathered from Republika, an Indonesian online newspaper. The method used to gather the data was “Focused Web Crawling” [26, 27]. “Focused Web Crawling” is a method to download or harvest particular data from websites, commonly from one website. The crawler was developed using Python programming language to crawl web addresses from the Republika website. There were 6111 articles downloaded, containing 6,947,178 words, 87,998 of which were unique.

B. Pre-Processing

Pre-processing is a required process to clean the dataset. Some steps in pre-processing are case folding, HTML tags removal, special characters removal, tokenizing, dealing with missing data, dealing with data error, and stemming. These stopwords extractions implement pre-processes are as follows:

- Case folding: Converting characters from uppercase to lowercase. The fastest and simplest way is entirely changing words to lowercase, including words occurring in a sub-title or title and words at the beginning of a sentence. Since some papers used uppercase for *Term Frequency* and others used lowercase *term frequency*, so in our research, we converted all words into lowercase, which means that we treat those two phrases as the same phrase.

- HTML tags removal: Removing all HTML tags, scripts, and other metadata from HTML documents is mandatory. It returns clean texts from documents in HTML format.

- Special characters removal: It includes removing punctuations, numbers, and other non-text characters. Examples of the special characters that removed from the text are ©%&+?,.:;-/'()[]{}\”\’=\"0123456789.

- Tokenizing: It separates each word from documents into an array of items or a bag of words.

C. Stopwords Extraction

Extracting stopwords from Indonesian documents is the primary purpose of this study. The stopwords extraction from the dataset used the TF-IDF method after the pre-processing steps. Eq. (1)-(5) present this TF-IDF:

$$TF - IDF(\omega_i) = tf(\omega_i) \times idf(\omega_i) \qquad (1)$$

$$tf(\omega_i) = \frac{n_{ij}}{\sum_{k=1}^{m} n_{kj}} \qquad (2)$$

$$idf(\omega_i) = log\left(\frac{N}{df(\omega_i)}\right) \qquad (3)$$

$$df(\omega_i) = \left|\{j: \omega_i \epsilon\, d_j\}\right| \qquad (4)$$

where $f(\omega_i)$ is frequency of occurrence of term or word ω_i in document j, and N is total number of all documents in document collection $\{d_j\}$. $df(\omega_i)$ indicates the number of documents contain term ω_i in the document collection. n_{ij} is the number of occurrences of ith term appearing in jth document. n_{kj} is occurrence frequency of kth term appearing in jth document. $|\{j: \omega_i \in d_j\}|$ is number of document consisting ith term. Getting the value of each term in every document is done by examining every term in the document collection or corpus.

For the whole document collection, corpus or dataset, the average of TF, $tf(\omega_i)$, is divided by the number of documents consisting of term ω_i. Thus, the TF-IDF formula of term ω_i for the whole document collection is:

$$TF - IDF(\omega_i) = \frac{\sum_j^N tf(\omega_{ij})}{df(\omega_i)} \times idf(\omega_i) \qquad (5)$$

IV. Experiments

Several experiments have been done to find the methods. For example, the data gathering method should be tried many times before we can harvest the data automatically. It is because the articles or documents are spread into tens of categories or sub-categories in the data source (https://www.republika.co.id/), such as News, Playing Games, Economics, Football, Islam Digest, or International. Since the structure of these web pages was not crawler friendly, we used Focused Web Crawling strategy to handle them. The data is then processed using the discussed pre-processing methods.

The pre-processing methods used were standard methods for Natural Language Processing. Our experiments regarding pre-processing proceeded smoothly. All pre-processes were done automatically by using applications developed in Python languages. The Python language was chosen because of its many machine learning libraries, especially for NLP. Later, a bag of words or an array of terms resulted from pre-processes fed into the TF-IDF method.

We used the TF-IDF formula shown in (5) to extract stopwords. Since there is no need for a training dataset, this NLP approach is categorized as an unsupervised machine learning method. It contains $idf(\omega_i)$ that comes from equation (3). It normalizes equation (5), limiting results of $TF - IDF(\omega_i)$ between zero and one.

If $TF - IDF(\omega_i)$ is equal to 0, it means that the i-th word (ω_i) exists in all documents. Table I shows three words contained in all documents that are *republikacoid* (republika.co.id), *wib* (west Indonesian time zone), and *lainnya* (others). The greater value of $TF - IDF(\omega_i)$ denotate that the word is less significant to be a stopword. Fig. 2 shows the correlation between TF-IDF$_{\mathrm{max}}$ and the number of stopwords extracted in the logarithmic scale. This figure shows that those words are stopwords if maximum of $TF - IDF(\omega_i)$ is equal to 1.

TABLE I. SAMPLE OF EXTRACTED STOPWORDS USING TWO DIFFERENT METHODS

No	Tala's & Wibisono's Method				This Research Method		
	Rank	*Term*	*Frequency*		*Rank*	*Term*	*TF-IDF*
1	1	yang	28,913		1	republikacoid	0
2	2	dan	27,074		2	wib	0
3	3	di	25,634		3	lainnya	0
4	4	untuk	13,760		4	terpopuler	0.0000007
5	5	dari	12,241		5	foto	0.0000007
6	6	dengan	11,839		6	terkait	0.0000011
7	7	pada	10,627		7	di	0.0000022
8	8	ini	10,478		8	home	0.0000039
9	9	photo	9,547		9	republikaid	0.0000044
10	10	dalam	8,525		10	copyright	0.0000045
11	17	republikacoid	6,116		11	reserved	0.0000045
12	68	newsroom@rolrepublikacoid	2,759		12	right	0.0000046
13	69	sekretariat@republikacoid	2,759		13	all	0.0000047
14	70	update	2,743		52	sekretariat@republikacoid	0.0002252
15	71	memicu	2,737		56	newsroom@rolrepublikacoid	0.0002282
16	72	marketing@republikacoid	2,734		57	marketing@republikacoid	0.0002282
17	73	kepada	2,727		58	us	0.0002283
18	74	direncanakan	2,703		59	gerai	0.0002289
19	75	tergolong	2,680		60	about	0.0002289
20	76	jis	2,674		61	copy	0.0002296

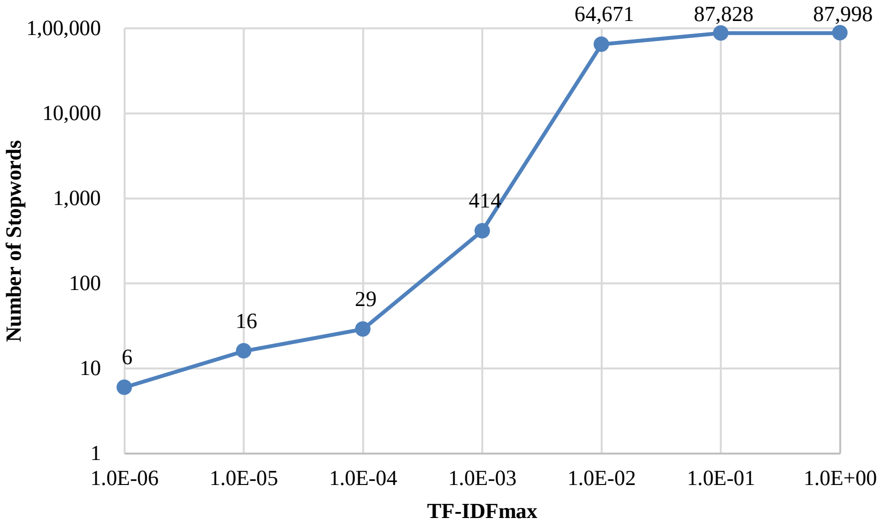

Fig. 2. The correlation of the threshold (TF-IDFmax) and the number of stopwords extracted in the logarithmic scale

VI. Results

The number of extracted stopwords using the proposed method depends on the defined TF-IDF threshold. For example, the system extracted only six stopwords for the maximum of TF-IDF 10^{-6}, and 414 stopwords if the maximum of TF-IDF increased to 10^{-3}. However, for TF-IDF$_{max}$ equal to 0.01, the number of stopwords is blown up to 64,571. The cut-off of the number of stopwords can be done by setting the value of TF-IDF$_{max}$. Since the range of TF-IDF is 0 to 1, the threshold can be maintained constantly. For example, if the threshold is set to 0.001 and the number of documents doubled the number of stopwords generated by TF-IDF does not change significantly. If we only use TF to extract stopwords and set the threshold to 8,000 and double the number of documents, then the frequency of stopwords generated might be doubled, resulting in the number of stopwords changing significantly as shown in Table I under Tala's and Wibisono's method.

The extracted stoplist contains some words specific to the dataset. For example, since the dataset or document collection source is Republika online daily newspaper, then there are some words with TF-IDF equal to zero. It means that those words occur in all documents. Table I shows the sample of extracted keywords from the same document collection using two different methods. Results in the left column are based on the previous researcher's method, and the right column is based on the method proposed in this work. As shown in this table, other methods can not reveal words that occurred in all documents.

Analyzing the top 100 extracted stopwords shows that the method used in this research, TF-IDF, is better than the previous methods. First, this research output can reveal the words that occur in all documents and place it in the top ranks, while the old method can reveal only two words and place them in ranks 17 and 26. Second, TF-IDF method can expose all words in the sentence “copyright … all right reserved” that occur in most of the documents, where the old method cannot reveal any of those words.

VII. Conclusions and Future Works

Stopwords extraction using TF-IDF has three advances compared to TF. First, it can detect words that occur in all documents with TF-IDF equal to zero. Second, it can implement threshold to limit the number of generated stopwords. Third, it can expose all words that occur in most of the documents and place it in the top ranks.

This research can be improved for future works in two ways. First, the documents in the corpus should be classified by its domain because stopwords for one domain are different from other domains. Secondly, develop a recommender system, a web-based application for the stopwords extraction that can be accessed by public.

Acknowledgment

Excellent Research Grants of Higher Education (PT-UPT), Directorate General of Higher Education, Research, and Technology, Ministry of Education, Culture, Research, and Technology of the Republic of Indonesia (Ditjen Dikti - Kemdikbud) titled "Representation of Multi Talents of Covid-19 Expert Based on indexed publication data, 2019 & 2020".

References

[1] J. Kaur and P. K. Buttar, "A systematic review on stopword removal algorithms," International Journal on Future Revolution in Computer Science & Communication Engineering, vol. 4, no. 4, pp. 207-210, 2018.

[2] M. Dehghani and M. Manthouri, "Semi-automatic detection of Persian stopwords using FastText library," in 9781665402088, 2021.

[3] S. Sahu and S. Pal, "Effect of stopwords in Indian language IR," Sadhana - Academy Proceedings in Engineering Sciences, vol. 47, no. 1, pp. -, 2022.

[4] A. Bichi, R. Samsudin and R. Hassan, "Automatic construction of generic stop words list for hausa text," Indonesian Journal of Electrical Engineering and Computer Science, vol. 25, no. 3, pp. 1501-1507, 2022.

[5] R. Arlitt, S. Khan and L. Blessing, "Feature engineering for design thinking assessment," in International Conference on Engineering Design, 2019.

[6] K. Goucher-Lambert and J. Cagan, "Crowdsourcing inspiration: using crowd generated inspirational stimuli to support designer ideation," Design Studies, vol. 61, pp. 1-29, 2019.

[7] H. Song, J. Evans and K. Fu, "An exploration-based approach to computationally supported design-by-analogy using D3," AI EDAM, vol. 34, pp. 444-457, 2020.

[8] S. Urologin, "Sentiment analysis, visualization and classification of summarized news articles: a novel approach," (IJACSA) International Journal of Advanced Computer Science and Applications,, vol. 9, no. 8, pp. 616-625, 2018.

[9] F. Shi, L. Chen, J. Han and P. Childs, "A data-driven text mining and semantic network analysis for design information retrieval," Journal of Mechanical Design, vol. 139, no. 11, 2017.

[10] B. Guda, B. K. Nuhu, J. Agajo and I. Aliyu, "Performance evaluation of keyword extraction techniques and stop word lists on speech-to-text corpus," International Arab Journal of Information Technology, vol. 20, no. 1, pp. 134-140, 2023.

[11] F. Z. Tala, "A study of stemming effects on information retrieval in bahasa Indonesia," Institute for Logic, Language and Computation Universiteit van Amsterdam The Netherlands, Amsterdam, The Netherlands, 2003.

[12] S. Wibisono and M. Utomo, "Dynamic stoplist generator from traditional Indonesian cuisine with statistical approach," Journal of Theoretical and Applied Information Technology, vol. 87, no. 1, pp. 92-98, 2016.

[13] S. Gandotra and B. Arora, "Automated stop-word list generation for dogri corpus," International Journal of Advanced Science and Technology, vol. 28, no. 19, pp. 884-889, 2019.

[14] K. Gorro, M. Ali, L. Lawas and A. Ilano, "Stop words detection using a long short term memory recurrent neural network," ACM International Conference Proceeding Series, pp. 199-202, 2021.

[15] Z. Nassr, N. Sael and F. Benabbou, "Generate a list of stop words in Moroccan dialect from social network data using word embedding," Ensa Marrakech;Faculte des Sciences et Techniques;Marrakech;Universite Abdelmalek Essaadi;Universite Cadi Ayyad, 2021.

[16] T. Kochhar and G. Goyal, "Design and implementation of stop words removal method for Punjabi language using finite automata," Lecture Notes on Data Engineering and Communications Technologies, vol. 106, pp. 89-98, 2022.

[17] G. Armano, F. Fanni and A. Giuliani, "Stopwords identification by means of characteristic and discriminant analysis," in 9789897580741, 2015.

[18] S. Gunasekara and P. Haddela, "Context aware stopwords for Sinhala text classification," in 9781538691366, 2018.

[19] Y. Nezu and T. Miura, "Extracting stopwords on social network service," in 9781643680446, 2019.

[20] S. Sarica and J. Luo, "Stopwords in technical language processing," PLoS ONE, vol. 16, no. 8 August, pp. -, 2021.

[21] F. Lan, "Research on text similarity measurement hybrid algorithm with term semantic information and TF-IDF method," Advances in Multimedia, vol. 2022, pp. -, 2022.

[22] J. ZHU, S. HUANG, Y. SHI, K. WU and Y. WANG, "A Method of k-means clustering based on TF-IDF for software requirements documents written in Chinese language," IEICE Transactions on Information and Systems, vol. 105, no. 4, pp. 736-754, 2022.

[23] L. Xiang, "Application of an improved TF-IDF method in literary text classification," Advances in Multimedia, vol. 2022, pp. -, 2022.

[24] H. Vranken and H. Alizadeh, "Detection of DGA-generated domain names with TF-IDF," Electronics (Switzerland), vol. 11, no. 3, pp. -, 2022.

[25] H. Toosi, M. Ghaaderi and Z. Shokrani, "Comparative study of academic research on project management in Iran and the world with text mining approach and TF–IDF method," Engineering, Construction and Architectural Management, vol. 29, no. 3, pp. 1553-1583, 2022.

[26] J. Liu, X. Li, Q. Zhang and G. Zhong, "A novel focused crawler combining web space evolution and domain ontology," Knowledge-Based Systems, vol. 243, pp. -, 2022.

[27] S. Rajiv and C. Navaneethan, "Hybrid gradient strategies in event focused web crawling," in 9781607685395, 2022.