The effect of stemming and removal of stopwords on the accuracy of sentiment analysis on indonesian-language texts

Aditya Wiha Pradana¹, Mardhiya Hayaty²
Universitas AMIKOM Yogyakarta, Indonesia

Abstract
Preprocessing is an essential task for sentiment analysis since textual information carries a lot of noisy and unstructured data. Both stemming and stopword removal are pretty popular preprocessing techniques for text classification. However, the prior research gives different results concerning the influence of both methods toward accuracy on sentiment classification. Therefore, this paper conducts further investigations about the effect of stemming and stopword removal on Indonesian language sentiment analysis. Furthermore, we propose four preprocessing conditions which are with using both stemming and stopword removal, without using stemming, without using stopword removal, and without using both. Support Vector Machine was used for the classification algorithm and TF-IDF as a weighting scheme. The result was evaluated using confusion matrix and k-fold cross-validation methods. The experiments result show that all accuracy did not improve and tends to decrease when performing stemming or stopword removal scenarios. This work concludes that the application of stemming and stopword removal technique does not significantly affect the accuracy of sentiment analysis in Indonesian text documents.

1. Introduction
Recently, the research of sentiment analysis on social media has attracted many researchers in the world. Such as is Twitter [1] as a microblogging tool allowing users free expression. Sentiment Analysis is used to find out the opinion about a topic which is as positive, negative, and neutral sentiments [2].

The word structure of comments on social media is irregular and contains much noise, and it is a challenge in conducting sentiment analysis [3][4], therefore the role of data preprocessing is essential because it can affect the accuracy and cannot be ignored when conducting sentiment analysis [4]. Preprocessing data is the process of cleaning and preparing data for review [5]. Preprocessing techniques for text classification are stemming and stopwords removal [6][7]. The "stemming" is turning a word into a root word by removing the phrase prefix [8]. While the "stopwords removal" is removed words that often appear and do not have any meaning [9].

Previous research used the TF and TF-IDF scenarios with the Naïve Bayes classification algorithm. Stemming had no significant effect on the classification accuracy of both the TF and TF-IDF scenarios [8]. Preprocessing in Arabic text using SVM algorithm has stated that normalization can increase efficiency from 96.66% to 97.50%, while "stemming" actually reduces accuracy using both ISRI Stemmer and Tashaphyne with an accuracy value of 93.06% and 95.83% [10]. Preprocessing text in English documents has been done and was concluded that using stemming and stopwords removal improves the accuracy of sentiment analysis in all situations [11].

However, whether stemming and removal of stopwords can also improve the accuracy of sentiment analysis in Indonesian documents given that a word has a different meaning in the language used. The purpose of this paper is to examine the effect of stemming and deletion of stopwords on the accuracy of sentiment analysis in Indonesian text documents-sentiment analysis using the SVM algorithm.

2. Research Method
In this section Figure 1, we briefly describe the experimental design. In the first step, we collect the data from twitter used GetOldTweets. In the second step, pre-processing; consist of stemming dan stopwords removal to clean the data from noise. Afterward, weighting TF-IDF. The classification process used Support Vector Machine (SVM), and the end prose evaluate the performance with confusion matrix and cross-validation.
2.1 Data Collecting
This study uses data from Twitter user comments collected using python program GetOldTweets. There are 2000 data tweets and labeled manually into two sentiment polarities, a positive and negative sentiment. With the number of positive tweets is 675, and 1325 negative tweets, shown the Table 1.

<table>
<thead>
<tr>
<th>Polarity</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>675</td>
</tr>
<tr>
<td>Negative</td>
<td>1325</td>
</tr>
</tbody>
</table>

2.2 Preprocessing
Preprocessing data is the process of cleaning and preparing data for analysis [5]. Preprocessing can also be used to reduce computational processes and feature space which can improve performance accuracy and classification. In the case of text classification, many preprocessing techniques can be used [12][13]. Preprocessing techniques used in this study are as follows.

a. Case folding aims to change all letters in a text document into lowercase letters [14].

b. Removing the URL. Many researchers argue that the URL does not carry information about the sentiment on Twitter [13].

c. Removing numbers, numbers on tweets has no effect on sentiment analysis, and deleting them can reduce noise and increase efficiency [15].

d. Removing punctuation, Punctuation is a unique character like an exclamation mark, comma, question mark, and others. It not required in sentiment classification [16].

e. Removing special characters from Twitter: cleanups such as deleting Twitter user usernames, hashtags, and non-ASCII characters [12].

f. Removing word less than three characters.

g. Normalization is the process of changing a word to standard form [17]. Normalization in this study also changed the slang word to ordinary word.

h. Stemming aims to turn a word into a root word by removing the phrase prefix and prefix [8]. This study uses Sastrawi Stemmer adapted from the Nazief-Andriani [18] algorithm with a modified confix-stripping [19].

i. Stopwords Removal is a word that often appears and does not have any meaning [9]. Stopwords in Indonesian such as "yang", "dl", "untuk", and "dar". In this study, the stopwords list used is Sastrawi stoplist.

j. Tokenization is a task to separate the full text string into list of separate words [4].

In this study, we conducted four models pre-processing, shown the Table 2.

<table>
<thead>
<tr>
<th>Preprocessing Model</th>
<th>Preprocessing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stem - Stop</td>
<td>Apply all pre-processing</td>
</tr>
<tr>
<td>No Stem</td>
<td>Without stemming</td>
</tr>
<tr>
<td>No Stop</td>
<td>Without stopwords removal</td>
</tr>
<tr>
<td>No Stem – No Stop</td>
<td>Without stemming and without stopwords removal</td>
</tr>
</tbody>
</table>

2.3 Weighting
The preprocessing stage finished, the next step is weighting uses Term Frequency - Inverse Document Frequency. TF-IDF reflects the importance of a word in a text document [20]. The Level of importance increases when a word appears several times in a document, but the frequency of words appearing a document keep balanced.

Term Frequency (TF) is the frequency with which words appear in a document. For term $t_i$ in a document, can be formulated follows Equation 1 [21].

$$t_{fi,j} = n_{i,j}$$ (1)
\( n_{i,j} \) is the number of occurrences of each word \( t_i \) on \( d_j \) document. Inverse Document Frequency (IDF) measures the general importance of a word in a document. I have formulated follows \textbf{Equation 2}.

\[
idf_{i,j} = \log \frac{D}{df_{i,j}}
\]  

\( D \) is the total number of text documents \( df_{i,j} \) is a number of document \( d_j \) which contains the term \( t_i \). TF-IDF is a combination of TF and IDF, the formula follows \textbf{Equation 3}.

\[
tf - idf_{i,j} = tf_{i,j} \times idf_{i,j}
\]  

\textbf{2.4 Classification}

Classification algorithm using the Support Vector Machine algorithm. SVM is a supervised machine learning algorithm, and this approach works if there are trained data and targeted data. The Support Vector Machine algorithm is a statistical classification approach based on maximizing the margin between instances and hyperplane separation [22]. SVM separate class of data by using three different lines, one for the main separating line and two other lines are supported line [23].

For example there is data training \( x \) and label \( y \in \{-1, 1\} \) to show the label class. Whereas -1 negative class and 1 is positive class. With the result that hyperplane formulas as \textbf{Equation 4} [22].

\[
w \cdot x + b = 0
\]  

In the equation above, \( w \) is the vector weight, and \( b \) is the bias factor. In Figure 2, two hyperplanes that determines the margin side. For the positive side hyperplane, the formula follows \textbf{Equation 5}.

\[
w \cdot x + b \geq +1
\]  

and negative side hyperplane, the formula follows \textbf{Equation 6}.

\[
w \cdot x + b \leq -1
\]  

while the distance between the two hyperplanes is shown on \textbf{Equation 7}.

\[
\frac{2}{||w||}
\]  

\textbf{2.5 Performance Evaluation}

The Evaluation to measure how appropriate the proposed method for classifying text. Evaluation using confusion matrix and k-fold cross-validation. Confusion matrix table for prediction of two classes as follows \textbf{Table 3} [12].

\textbf{Figure 2. Support Vector Machine Hyperplanes}
Table 3. Confusion Matrix

<table>
<thead>
<tr>
<th>Predicted Class</th>
<th>Actual Class</th>
<th>Class-1 True Positive (TP)</th>
<th>Class-2 False Positive (FP)</th>
<th>False Negative (FN)</th>
<th>True Negative (TN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class-1</td>
<td>Class-1</td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
<tr>
<td>Class-2</td>
<td>Class-2</td>
<td>False Positive (FP)</td>
<td>True Negative (TN)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The confusion matrix table above is used to calculate the accuracy of the proposed method. The formula calculates the accuracy follows Equation 8.

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\] (8)

Afterward, further evaluation uses k-fold cross-validation. The method works are to divide the dataset randomly as many as "k" separate parts of the same size, and each piece is used to test the model with a classification algorithm [24]. Calculation of k-fold cross-validation produces average accuracy. In this study, the value of "k" is 10.

3. Results and Discussion

This study uses python programming and machine learning tools called scikit-learn to conduct all the experiments. Moreover, this research tested classification performance 35 times for each preprocessing approach that we proposed for this study and using the average accuracy to compare each other's performance.

The experiments result using both of the confusion matrices, and k-fold cross-validation is shown in Figure 3. The best accuracy of the confusion matrix test used the stemming with an accuracy score of 81.44%. While the stopword removal implementation got the worst accuracy score of 80.51%, the other's are both stemming and stopword elimination is 81.3% and without implementing both 81.4%.

On the other hand, the k-fold cross-validation test shown the different results in terms of the best accuracy with accuracy score was 81.06% on without both stemming and stopwords removal. While the lowest accuracy score is 80.26% with stopword elimination and the others are 80.71% and 81.05%. The result of both the confusion matrix and k-fold cross validation has shown there are no significant differences between all preprocessing model. The difference between the highest and the lowest accuracy are only 0.93% and 0.8%.

Compare to the prior research [8], and There is very lightly enhancement regarding the accuracy differences with the same situation. The effect of stemming technique implementation, which in this experiment got 0.93% better while the prior research reduces the accuracy by 1.34% when stemming technique is applied.

Accuracy does not increase significantly because the way stemming works only cuts a word into a basic word so that sometimes it has a misunderstanding, whereas, for sentiment analysis, the meaning of the word has a critical role in judging an opinion person.
4. Conclusion

This paper examines the effect of stemming and stopword removal implementation on the preprocessing step toward the accuracy of sentiment analysis in Indonesian text documents. The result, the application of the stemming and stopword removal technique on the preprocessing stage does not significantly affect the accuracy of sentiment analysis with the accuracy differences of the highest and the worst are only 0.93% and 0.8% on both confusion matrix and k-fold cross-validation test.

In future work, we suggest trying the lemmatization technique for conducting sentiment analysis in which the lemmatization does full morphological analysis to identify the root word for each word accurately.

References


