A Comparative Study of Sentiment Analysis using SVM and SentiWordNet

Mohammad Fikri
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Jawa Timur
fikri.mohammad15@mhs.if.its.ac.id

Abstract— **Sentiment analysis has grown rapidly which impact on the number of services using the internet popping up in Indonesia. In this research, the sentiment analysis uses the rule-based method with the help of SentiWordNet and SVM algorithm with TF-IDF as feature extraction method. Since the number of sentences in positive, negative and neutral classes are imbalanced, the oversampling method is implemented. For imbalanced dataset, the rule-based SentiWordNet and SVM algorithm achieve accuracies of 56% and 76%, respectively. However, for the balanced dataset, the rule-based SentiWordNet and SVM algorithm achieve accuracies of 52% and 89%, respectively.**

Keywords-sentiment analysis; sentiwordnet; wordnet; rule-based; support vector machine

I. INTRODUCTION

The development of information technology in Indonesia can be spelled out very rapidly which has an impact on the increasing number of internet users in Indonesia. Based on data from Internet World Stats in Q4 2017, internet users in Indonesia amounted to 143.26 million or about 53.7% of the Indonesian population. Currently, almost every information is available on the Internet. Even every activity that we do can be helped by using the internet, such as shopping everyday needs, order transportation tickets, order food or used as a medium of communication between communities.

With the increase in the number of Internet users can open an opportunity to give good impact to an organization. Due to the data generated due to internet user activity. These data can be opinions or facts to something. In this paper focus on public opinion about products in the form of applications on smartphones. These opinions can be further analyzed which will result from the analysis used as a consideration of a company's decision-making that creates the application.

Among the various technical analyzes, the technique is called sentiment analysis [1]. This technique is used to process text documents in the form of opinions that later generated an information so that opinions can be distinguished into a positive, negative, or neutral opinion.

In the development of information technology, opinion mining is one of the interesting research topics in the field of Natural Language Processing (NLP).

In this paper, we compared the implementation of supervised machine learning and rule-based for sentiment analysis. For implementations using supervised machine learning, we use TF-IDF to convert text into classifiable features and for classification processes using Support Vector Machines, Unfortunately, SentiWordNet currently does not support languages other than English. So the first thing is we have to translate the opinion into English so that they can be done the analysis of the opinion. The data is taken from Google Play store and Apple Appstore. The method to get the data is the same as in the paper [2], [3]. The sentiment analysis technique we use lexicon-based. A sentence is split into vocabulary or commonly referred to as unigram.

This paper consists of: Section 2 contains the same research that has been done on sentiment analysis; Section 3 contains an explanation of the method used; Section 4 contains an explanation of the results of the analysis; Section 5 contains the conclusions.

II. RELATED WORKS

The development of research on sentiment analysis using other than English has been widely practiced.

Farra et al. [4] conducting research on opinion mining on public opinion about films using Arabic as its main language. The method used is the first public opinion on English-language films translated into Arabic. Then sentiment analysis using SVM method (Support Vector Machine).

Denecke [5] researches public opinion on films using German as the main language. The first sentence method is translated into English using PROMPT XT as a technology to translate from German to English and then preprocessing where the word stop-word is erased and every word is returned to the stemmed form. The sentence was analyzed using SentiWordnet.

For those who use Indonesian as their primary language, Naradhipa [6] conducts research on public opinion contained in social media. The first method is to do preprocessing which eliminates punctuation when used excessively like "Well ...." To "Yah", replacing the informal abbreviation into a more formal word like "q love right?" To "I love you?", and fix if there are errors in words like "SEMANGATTTT" to "SPIRIT" and remove stop-word. Then do the comparison between classification algorithm using SVM, Maximum Entropy, and Rule-based method.

Sun et al. [7] conduct research comparing several methods of text classification with unequal datasets between positive and

ISBN 978-602-53524-0-9

negative so that data must be under-sampled in the majority class and synthesize new data in minority classes. Then do the classification using SVM method.

III. RESEARCH METHODS

The main objective of this study is to compare text classification algorithms between using a rule-based algorithm with the help of SentiWordNet and using TF-IDF as a feature extraction algorithm from text to a vector and then classified using SVM algorithm to an imbalanced dataset on the number of positive, negative, and neutral classes.

The results of each algorithm used will be sorted and compared based on which has the highest scores on the F-score and accuracy.

Figure 1. Research Overview Diagram

A. Data Construction

The dataset contains public opinions about some apps taken from Google PlayStore and Apple AppStore (https://github.com/oklahomaok/AppStoreReview). The data consists of “id_komen”, “title_komen”, and “komen”. The sentiments for each sentence are determined by humans into three classes, which are positive, neutral, and negative. The dataset contains 553 sentences, which are 259 positive class, 241 negative class, and 53 neutral class.

The positive class and negative class are balanced, however, the neutral class is imbalanced because the neutral class has sentences much less than the others. The data is stored in a data frame [8] with the "COMMENT" column for comment, "SENTIMENT" column for the correct sentiment, and "SENTIMENT_ID" column for sentiment id where "0" for negative, "1" for positive, "2" for neutral.

B. Balancing Data

Balancing an unbalanced dataset is a critical process in machine learning [7]. The method used this time by oversampling the minority class [7]. With the following steps :

Figure 2. Balancing Imbalanced Dataset Process

The explanations of Fig. 2 are as follows :

1) Marking The Minority Class and Majority Class. Create one column named "flag_balance". Mark the minority class (neutral) by filling in the "flag_balance" field with 1 and the majority class (positive and negative) with 0.
2) Splits Into 2 Dataframes.
   The data which columns "flag_balance" are 1 becomes the minority data frame [8] and the data which column "flag_balance" are 0 become the majority data frame [8].
3) Resample The Minority Class Dataframe.
   By using the existing algorithm in the scikit-learn to oversampling the minority class that is resampled [7], [9]. Resample randomly until the number of minority classes equals the average number of majority classes. In this paper, neutral is the minority class, positive and negative is the majority class.
4) Combine The Majority Class Dataframe and The Upsampled Minority.
   Merge both data frame (majority and minority) using CONCAT function [8], random the sequence on the data frame so that the two data have been merged into random.

C. Preprocessing Data

The pre-process is done to change the text into its formal form.

With the following preprocessing steps below.

1) “Enter” Character Normalization.
   Remove "\ n" or enter on the sentence to be a single line only.
2) Lowercase Normalization.
   Turn the sentence into all lowercase.
3) Unnecessary Character Normalization.
   Delete a recurring character whether it's an alphabet or a punctuation as done by Naradhipa [6]. For example "Setiaaaa" to "Setia" and “Yah....." to "Yah".
4) Punctuation Normalization.
   Delete punctuation on the sentence.
5) Slang Word Normalization.
   Fixed informal words and abbreviations. The way it is done by using manual way, not using spellchecking, the manual way here means matching the word with a hash map containing slang word if the word matches the

ISBN 978-602-53524-0-9

key of the slang word hashmap then the word will be changed to the value of the key of the hash map.

For example, abbreviations such as "spt" to be "seperti" and informal words like "pake" to "pakai".

6) Stopword Removal

Delete the words that often appear in each sentence. The type of the word that is deleted is a conjunction word. For example, "dan", "serta", "serta", and others.

Below, Table I is an example of the preprocessing results.

TABLE I. PREPROCESSING RESULTS

No	Original Text	After Preprocessing Text
1	BETULIN DONG APLIKASINYA, RUSAK MULU NIH!!!!	betulin dong aplikasinya rusak melulu nih
2	App nya crash terus!! Tolong diperbaiki agar service nya semakin baik	app nya crash terus tolong diperbaiki service nya semakin baik

D. Rule-Based Using SentiWordNet

Because the purpose of this research is to compare two different methods and one of them is to use SentiWordNet so that the process of classification will also be different, especially since SentiWordNet is currently very limited and not yet available in Bahasa Indonesia.

1) Translate Data

Because Sentiwordnet is currently still limited to the Indonesian language therefore from this process the data is translated into English. For the translation process, Google Translate is used as the language translator tool.

The results of this translator tool can be assumed quite good although there are still some sentences that do not have the correct sentence structure.

2) Tokenization and POS Tagging

Tokenization is a process to split one sentence into a piece of the word. At this process, the sentence is split into unigram which means the sentence is split into several parts consisting of 1 piece of a word using NLTK word_tokenize function [10].

After the tokenization process, every 1 piece of word is determined what part of speech of the word using NLTK pos_tag function [10]. There are 8 parts of speech which are nouns, pronouns, adjectives, verbs, adverbs, prepositions, conjunctions, and interjections.

However, the part of speech tag is Penn Treebank POS tag so that because SentiWordNet is used for this research and SentiWordNet only have four general POS tags of noun (N), verb (V), adjective (A), and adverb (R), so converting the POS tag to SentiWordNet POS tags is necessary [11] with the following rules below.

a) Noun (N)

If POS tags are 'NN', 'NNS', 'NNP', 'NNPS' then it should be changed to 'N'.

b) Verb (V)

If POS tags are 'VB', 'VBD', 'VBG', 'VBN', ‘VBP' or ‘VBZ' then it should be changed to 'V'.

c) Adjective (A)

If POS tags are 'JJ', 'JJR', or 'JJS', then it should be changed to 'A'.

d) Adverb (R)

If POS tags are 'RB', 'RBR', or 'RBS', then it should be changed to 'R'.

The latter on this process is done lemmatization, lemmatization is a process where a word is returned to its basic form back in accordance with the POS tag. This process is done using the NLTK lemmatizer function [10].

3) Sentiment Classification

The sentiment classification in this study uses SentiWordnet and Wordnet tools. SentiWordnet is used to find the score of each synset and Wordnet is used to search for synonyms of each word being analyzed.

Scores for each word are searched using Sentiwordnet according to POS tags if the scores are more than 0 then it will be taken, otherwise it will be bypassed.

Then sentiment score per word has been obtained, we have to do a total calculation to get sentiment score for one sentence. The semantic orientation calculation uses the method according to Formula (1) and (2).

$$= \sum \in \quad (1)$$

$$= \sum \in \quad (2)$$

Formula (1), (2), $Score_{positive}$ is the final number of positive scores while the $Score_{negative}$ is the final number of negative scores. And n is the number of words whose

first sentence value is above 0. Then to get sentiment value if the sentence is positive, negative or neutral using Formula (3).

$$\begin{cases} - & \geq 0.05 \\ - & \leq -0.05 \\ -0.05 < & - < 0.05 \end{cases} \quad (3)$$

Sentiment value obtained using positive score difference and negative score. If the score difference is greater than 0.05 then the sentiment value is positive. If the score difference is smaller equal to -0.05 then the sentiment value is negative. And if the score difference is smaller than 0.05 and greater than -0.05 then the sentiment value is neutral.

E. Supervised Machine Learning using SVM

In this section, the TF-IDF method is used as the feature extraction process from text to vector and SVM is used as an algorithm for text classification.

1) Feature Extraction using TF-IDF

Term Frequency-Inverse Document Frequency is a method for converting a document (sentence) in a corpus into a statistically measurable weight in which this weight represents how important the word is in the document or phrase [12].

Here is how to transform a corpus into a weight using the TF-IDF method.

a) Tokenization

ISBN 978-602-53524-0-9

Documents that exist in a corpus should be tokenized into unigram and bigram. Unigram consists of one word and Bigram consists of 2 words. The example can be seen in Fig. 3.

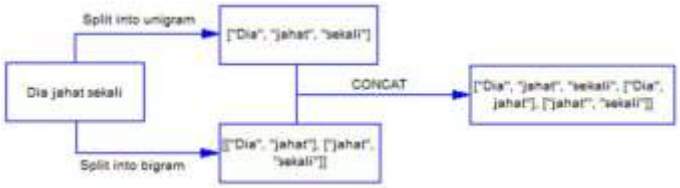

Figure 3. Tokenization Process

b) Term Frequencies

Term Frequencies measure how often a word appears in a document [12]. It is possible that a term would appear much more times in long documents than shorter ones. Term Frequencies is the total count of a term in a document.

c) Inverse Document Frequency

Inverse Document Frequency measures how important a term to a document. When we calculate term frequencies, we assume that all terms have the same importance in a document. Whereas conjunctional words such as "dan", "adalah", and "serta" very often appear in several documents (sentences) thereby reducing how important the word is in a sentence.

$$(,) = \ln \frac{( )}{((,))} + 1 \quad (4)$$

Formula (4), IDF (t,d) is the inverse document frequencies of a term in a document, n is the total number of the documents, df(t,d) is the number of documents with a term (t) in it.

The effect of adding "1" to the idf in the equation above is that terms with zero idf, i.e., terms that occur in all documents in a training set, will not be entirely ignored [9].

The constant "1" is added to the numerator and denominator of the idf as if an extra document was seen containing every term in the collection exactly once, which prevents zero divisions [9].

d) Calculate TF-IDF Weight

In the process of calculating weights using the TF-IDF method, the sci-kit learn tools with TfidfVectorizer class [9] is implemented where all the formulas used are in accordance with Formula (4), (5). And in this section will be exemplified how the calculation of weights using the TF-IDF method.

$$(,) = (,) \quad (,) \quad (5)$$

e) Normalize TF-IDF Weight

Normalization is done so that the tf-idf weights on the same term but is on a different document and the document has a very different sentence length that has a well-balanced weight. Normalization is done using L2 norm so that the weight of tf-idf for each term has a weight of 0-1 scale, see Formula (6).

$$= \frac{}{\| \|} = \frac{}{\sqrt{\quad + \quad + \quad + \ldots}} \quad (6)$$

Below is an example of text that will put into TF-IDF weighting in Table II and Table III.
D1 = "dia baik sekali."
D2 = "dia jahat sekali."

TABLE II. TF-IDF WEIGHTING

Term	Tf		Df	Idf
	T1	T2		
dia	1	1	2	1
baik	1	0	1	1.4054651
sekali	1	1	2	1
jahat	0	1	1	1.4054651
dia baik	1	0	1	1.4054651
baik sekali	1	0	1	1.4054651
dia jahat	0	1	1	1.4054651
jahat sekali	0	1	1	1.4054651

TABLE III. TF-IDF WEIGHTING

Tf-Idf		Tf-Idf (L2 Norm)	
T1	T2	T1	T2
1	1	0.3552001	0.355200085
1.4054651	0	0.4992213	0
1	1	0.3552001	0.355200085
0	1.4054651	0	0.499221327
1.4054651	0	0.4992213	0
1.4054651	0	0.4992213	0
0	1.4054651	0	0.499221327
0	1.4054651	0	0.499221327

2) Support Vector Machine

Support Vector Machine (SVM) is a classification method for linear or nonlinear data by using nonlinear data mapped to convert training data to a higher dimension. This method will find hyperplane by maximizing margin or distance between classes [7], [13], [14].

Considering the class in classification, the one vs rest strategy is implemented, this strategy consists in fitting one classifier per class. For the implementation, scikit-learn LinearSVC class [9] is used.

F. Comparing Results

Results from the classification of rule-based using SentiWordNet and supervised machine learning, SVM algorithm with TF-IDF as feature extraction compared with recall, precision, f1-score parameters.

Precision is the ability of a classification model to identify only the relevant data points [9], [14], see Formula (7). Recall is the

ISBN 978-602-53524-0-9

ability of a model to find all the relevant cases within a dataset [9], [14], see Formula (8). F1-score is the harmonic mean of precision and recall [9], [14] taking both metrics into account in the Formula (9). Accuracy is the quality or state of being correct or precise [14], see Formula (10).

$$= \frac{}{(\quad)} \tag{7}$$

$$= \frac{}{(\quad)} \tag{8}$$

$$1 \quad = 2 \frac{(\quad)}{(\quad)} \tag{9}$$

$$= \frac{}{} \tag{10}$$

Then for the split between training data and data testing using K-Fold Cross Validation method. So the data is divided into k section and then will be executed classification process as much as the k and for the data, testing is selected from one of k fold and training data is fold which not used as data testing [13], [14]. The selection of data testing per round is selected in sequence starting from the folds 1, see Fig. 4 for the illustration. In this study, scikit-learn K-Fold class [9] is implemented for the splitting process. For example, round 1 is used as data testing fold 1 and so on.

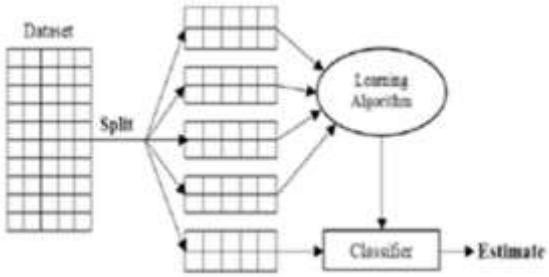

Figure 4. K-Fold Cross-Validation

IV. Results and Analysis

Implementation of this research is created by using Python Programming Language. The total number of the dataset is 553 data with detail for positive class 259 data, negative class 241 data, and neutral class 53 data. Then splitting between data training and data testing, we set k = 10 for the k-fold cross-validation splitting method.

```
Fold 10
Jumlah data training positif :  217
Jumlah data training negatif :  228
Jumlah data training neutral :  53
```

Figure 5. Underfitting on Neutral Class

For the 10th round k-fold cross-validation, the data testing for the neutral class does not exist at all because all of the 53 neutral class data has become training data, see Fig. 5.

Due to the underfitting dataset, the dataset is balanced using our proposed method. During the process of balancing the dataset, the amount of neutral class data increases to 250 data due to the average of positive and negative data.

TABLE IV. Comparison of Results using 10-Fold Cross Validation

	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
SVM without Balancing Method	48.74	53.23	50.89	75.75
Rule-based SentiWordNet without Balancing Method	49.5	46.42	47.76	55.81
SVM with Balancing Method	82.02	85.45	83.69	89.06
Rule-based SentiWordNet with Balancing Method	51.34	49.65	50.45	51.59

The results with the balancing method shown in Table 3 show that the SVM algorithm with the TF-IDF method as feature extraction is superior with f-score 83% and accuracy 89% compared to using rule-based SentiWordNet which get f-score 50% and accuracy 51%.

The results from Table IV can be compared, the balanced datasets get better results if using SVM algorithm with TF-IDF as feature extractor increases the accuracy and f-score because in particular the datasets which are neutral class has been balanced in number but the SentiWordNet rule-based algorithm has decreased both in accuracy and f-score.

We analyzed for using the rule-based SentiWordNet method found the average number of words not found synsets totaling 573 words. Whereas if the word amounting to 573 found synsets can increase the accuracy to about 20%.

V. Conclusion

Based on the results of the classification using SVM and rule-based, we can be concluded that :

1. Balancing datasets can improve both accuracy and f-score for SVM algorithm with TF-IDF as feature extraction method but can decrease both accuracy and f-score for the ruled-based SentiWordNet method.
2. SVM algorithm with TF-IDF as feature extraction method is much better to use compared to rule-based SentiWordNet.
3. There are still many words that do not have synset because Indonesian vocabulary is still incomplete if using SentiWordNet and tools translator is still not good in translating Indonesian to English in language structure.

ISBN 978-602-53524-0-9

REFERENCES

[1] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," Found. Trends® InformatioPang, B., Lee, L. (2006). Opin. Min. Sentim. Anal. Found. Trends® Inf. Retrieval, 1(2), 91–231. doi10.1561/1500000001n Retr., vol. 1, no. 2, pp. 91–231, 2006.

[2] M. R. Islam, "Numeric rating of Apps on Google Play Store by sentiment analysis on user reviews," 1st Int. Conf. Electr. Eng. Inf. Commun. Technol. ICEEICT 2014, pp. 1–4, 2014.

[3] E. Guzman and W. Maalej, "How do users like this feature? A fine grained sentiment analysis of App reviews," 2014 IEEE 22nd Int. Requir. Eng. Conf. RE 2014 - Proc., pp. 153– 162, 2014.

[4] N. Farra, E. Challita, R. A. Assi, and H. Hajj, "Sentence-level and document-level sentiment mining for arabic texts," Proc. - IEEE Int. Conf. Data Mining, ICDM, pp. 1114–1119, 2010.

[5] K. Denecke, "Using SentiWordNet for multilingual sentiment analysis," Proc. - Int. Conf. Data Eng., pp. 507–512, 2008.

[6] A. R. Naradhipa and A. Purwarianti, "Sentiment Classification for Indonesian Messages in Social Media," Int. Conf. Electr. Eng. Informatics, no. July, pp. 2–5, 2011.

[7] A. Sun, E. P. Lim, and Y. Liu, "On strategies for imbalanced text classification using SVM: A comparative study," Decis. Support Syst., vol. 48, no. 1, pp. 191–201, 2009.

[8] W. McKinney, "Data Structures for Statistical Computing in Python," Proc. 9th Python Sci. Conf., vol. 1697900, no. Scipy, pp. 51–56, 2010.

[9] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[10] S. Bird, E. Klein, and E. Loper, Natural language processing with Python. 2009.

[11] B. S. Rintyarna and R. Sarno, "Adapted weighted graph for Word Sense Disambiguation," 2016 4th Int. Conf. Inf. Commun. Technol. ICoICT 2016, vol. 4, no. c, 2016.

[12] H. C. Wu, R. W. P. Luk, K. F. Wong, and K. L. Kwok, "Interpreting TF-IDF term weights as making relevance decisions," ACM Trans. Inf. Syst., vol. 26, no. 3, pp. 1–37, 2008.

[13] B. Y. Pratama and R. Sarno, "Personality classification based on Twitter text using Naive Bayes, KNN and SVM," Proc. 2015 Int. Conf. Data Softw. Eng. ICODSE 2015, pp. 170–174, 2016.

[1] [14] M. Jupri and R. Sarno, "Taxpayer compliance classification using C4.5, SVM, KNN, Naive Bayes and MLP," in 2018 International Conference on Information and Communications Technology (ICOIACT), 2018, pp. 297– 303.

ISBN 978-602-53524-0-9