Optimizing the Hyperparameter of Feature Extraction and Machine Learning Classification Algorithms

Sani Muhammad Isa[1], Rizaldi Suwandi[2], Yosefina Pricilia Andrean[3]
Computer Science Department,
BINUS Graduate Program Master of Computer Science
Bina Nusantara University Jakarta,
Indonesia 11480

***Abstract*—The process of assigning a quantitative value to a piece of text expressing a mood or effect is called Sentiment analysis. Comparison of several machine learning, feature extraction approaches, and parameter optimization was done to achieve the best accuracy. This paper proposes an approach to extracting comparison value of sentiment review using three features extraction: Word2vec, Doc2vec, Terms Frequency-Inverse Document Frequency (TF-IDF) with machine learning classification algorithms, such as Support Vector Machine (SVM), Naive Bayes and Decision Tree. Grid search algorithm is used to optimize the feature extraction and classifier parameter. The performance of these classification algorithms is evaluated based on accuracy. The approach that is used in this research succeeded to increase the classification accuracy for all feature extractions and classifiers using grid search hyperparameter optimization on varied pre-processed data.**

Keywords—Sentiment analysis; word2vec; TF-IDF (terms frequency-inverse document frequency); Doc2vec; grid search

I. INTRODUCTION

Google Play is an online service that developed and operated by Google. This is an official app store for Android-based mobile phone. The entire google play service can be accessed through the Play Store app. Google Play sells Android apps, games, movies, and even e-books. Google play store apps data chosen because have enormous potential to drive app-making business to success. This research focused on apps review.

In recent years, the opinions of friends, domain experts are our consideration for decision making in our life. For example, which app is best to download, games to play, or e-book to read. Sentiment analysis or opinion mining plays an important role in this process [1]. The sentiment (expressions) states in natural language form.

With the increasing development in e-commerce, the needed to extract valuable information from consumer comment also increasing. It is important for the organization (Google Play developer company) to automatically identify each customer review whether it is positive, negative, or neutral [2]. The product comments contain a wealth of information about product evaluation from customers [3]. With the main form of information from the internet is text [4], text processing is needed the most. Text processing is needed for extracting the value of sentiment review.

Text classification research started from design the best feature extraction method to choose the best classifiers. Almost all techniques of text classification based on words [5]. For sentiment classification, this paper uses a machine learning based method because it has been widely adopted due to their excellent performance.

Word2vec, Doc2vec, and Terms Frequency-Inverse Document Frequency (TF-IDF) feature extractions that used in this research were implemented by python algorithm using the Sklearn library (TF-IDF) and the Gensim library (Word2vec & Doc2vec). Word2vec is a new open source feature extraction method based on deep learning [3]. Word2vec can learn the word vector representation and calculate the cosine distance in the high dimensional vector space. This research used word2vec because this approach can find the semantic relationships between words in the document.

TF-IDF is very important in this research. Balancing the weight between the less commonly used words and most frequent or general words is one of the capabilities of TF-IDF feature extraction. TF-IDF can calculate the frequency of each token in the review. This frequency shows the importance of a token to a document in the corpus [6].

To extend the learning of embeddings from word to word sequences, this research uses a Doc2vec as a simple extension to Word2vec. Many types of texts used this feature. They are word n-gram, sentence, paragraphs or document [7]. Refer to the embedding of the word sequence; we used the term document embedding that supported by Doc2vec.

The classification method was done with 3 classifier NB (Naive Bayes), SVM (Support Vector Machine), and DT (Decision Tree). SVM can be used to create the highest accuracy results in text classification problems [1]. The NB has high accuracy than other followed by the DT classifier.

This research dedicated to select the best feature extraction and choosing the best model for multiclass classification by comparing the TF-IDF, Word2vec, Doc2vec feature extraction and increase the accuracy using hyperparameter optimization. Hyperparameter optimization used to search the best parameter that produces the best classification accuracy. To selects, a point in space (in linear or log space) hyperparameter space using grid search is suitable in this case. Hyperparameter tuning is well-suited to use in some

derivative-free optimization, it is reflecting characteristic grid search to solve this problem [8].

This paper consisted of several parts of the section, there are: Section II to gives literature review, Section III describes the methodology of some techniques used in the research, Section IV describes result and analysis in this research, and Section V will conclude the paper.

II. Related Work

To know what is the mood or effect from text expressing, it can use sentiment analysis with assigning a quantitative value (positive, negative, and neutral). Previous research has shown that sentiment analysis has a good accuracy, such as, Twitter [2], application reviews [9], documents [10], texts [3], [4], [11], newsgroup [12], and news article [13], IMDB [14]. The Google Play review dataset from Kaggle was used in this research. Kaggle is an online web service that provides a series of a dataset that can be used to research. Data has filtered for noise and null review user's data also contain the sentiment for each review.

Sentiment analysis is a good candidate for analyzing sentiments in text classification. Using machine learning, the classification of documents was done. This machine learning automatically classifies the document into categories that have been labeled before. It can see as a supervised learning task because of the objective [12].

In Table I, there is attached some research about sentiment analysis from Google Scholar. This literature search using some keywords such as "Sentiment Analysis using Word2vec and TF-IDF", "Sentiment Analysis Google Play Review", "Sentiment Analysis using Doc2vec".

Based on the literature review, most of the research is focused on getting better accuracy. The method was designed according to the characteristics of the text. To represent the rank among the best approach in retrieving documents and labeling document, TF-IDF was used, Word2vec to obtain more accurate word vector, Doc2vec is one of the easiest ways is using an average of all words in the document to represent the feature of this document [18].

SVM, NB, and DT classifier were used in this paper for text classifier. In the classification process, many classification methods and machine learning techniques have been used. Using machine learning can increase accuracy by using optimization algorithm, i.e. hyperparameter optimization using a grid search. By adding a hyperparameter optimization, will get a better result with to determine hyperparameter efficiency in choosing parameter [19]. The methods have been used by [12] and [4] both methods produce a high level of accuracy with more than 80%. Then using this method will get a high accuracy [16].

TABLE I. Literature Review

Author	**Dataset**	**Method**	**Result**
J. H. Lau and T. Baldwin [7]	Document	Doc2vec	Better accuracy
R. Ju, P. Zhou, C. H. Li, and L. Liu [15]	Newsgroup	Latent Semantic Analysis + Word2vec	Better accuracy
D. Zhang, H. Xu, Z. Su, and Y. Xu [3]	Text (Chinese comments)	Word2vec and SVM-perf	Excellence accuracy
J. Lilleberg, Y. Zhu, and Y. Zhang [12]	Newsgroup	Word2vec, Terms Frequency-Inverse Document Frequency	Word2vec is the best solution
D. Rahmawati and M. L. Khodra [13]	Article	Word2vec	Better accuracy
S. K. R. Abinash Tripathy, Ankit Agrawal [14]	IMDb	Naïve Bayes, Max Entropy, SVM, SGD	Get better accuracy
P. Vateekul and T. Koomsubha [2]	Twitter	Long Short Term Memory and Dynamic Convolutional Neural Network	Better than Naïve Bayes and SVM
W. Zhu, W. Zhang, G.-Z. Li, C. He, and L. Zhang [16]	Text	Word2vec and Terms Frequency-Inverse Document Frequency	Better than Latent Semantic Analysis and Doc2vec
Y. Xi. Jin Gao, Yahao He, Xiaoyan Zhang [4]	Text	Word2vec, Terms Frequency-Inverse Document Frequency	Comparison
S. Fujita [17]	Newspaper	Contextual Specificity Similarity, K-Nearest	Contextual Specificity Similarity good in long text
L. Lin, X. Linlong, J. Wenzhen, Z. Hong, and Y. Guocai [11]	Text	CD_STR, TF-IDF weighted vector space model	Comparison
Q. Shuai, Y. Huang, L. Jin, and L. Pang [18]	Review	Doc2vec, Support Vector Machine, LogReg	Support Vector Machine, Logreg show a better result

III. METHODOLOGY

In the classification task using machine learning, the main important thing is the feature selection [18]. This research using 10.000 data of Google Apps from Kaggle. The reviews are divided into three class categories: positive, neutral and negative. The data then divided into train and test data with 80% of train data. The research methodology can be represented in Fig. 1.

A. Dataset

In this paper, dataset consisted of user reviews from Google Play complete with the sentiment analysis (Positive, Neutral, and Negative) for each review. Google Play stores allow users to write reviews about the downloaded applications. The data set contains 64,294 reviews, consists of the name application, review, sentiment, sentiment popularity, and sentiment subjectivity. This research used 10.000 reviews, it is consists of name applications, sentiments, and reviews because wanted to evaluate our approach against reviews containing diverse vocabularies, and sentiment as parameter classification. From the original, data are filtered by removing noise and null data using python NLTK library. Moreover, words that typo also deleted and modified the spelling of words using class Spelling Replacer.

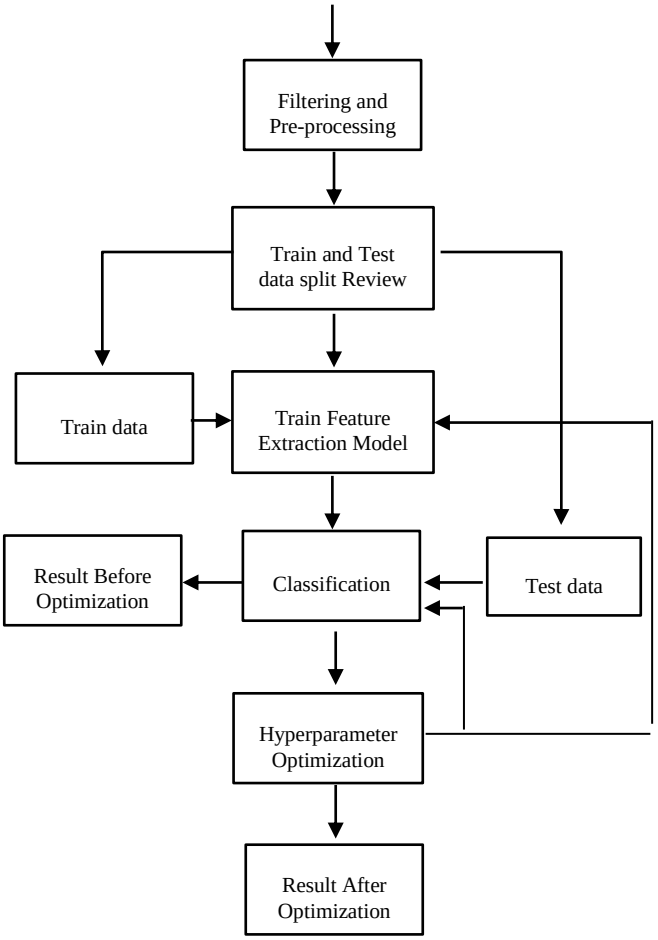

Fig. 1. Experiment Process.

B. Filtering

Preprocessed reviews were done by these techniques

1) Stopword removal: To eliminate term that is very common in the English language, we delete the stopwords (e.g., “such”, “was”, “any”, ”then”, etc). By using library NLTK corpus, the words we added to the stopwords list are “ ‘m ” and “ ‘s ”, also deleted the stopwords list are “y”, “o”, “s”, “i”, “d”. We remove punctuation at the end of the sentence (in the end the alphabet (upper & lower letters) or punctuation (@#$%^&*()-_=+~!`{[}]|\:;"<,>.?/).

2) Spelling replace: We use spell method from the autocorrect library to checking out all of the words. This library is able to correct the words which are not in accordance with the English word dictionary. For example, the word before corrected is “lke” and after corrected will be “like”.

3) Noise & null removal: We remove a column if the data has an empty row from name application, review, and sentiment. Moreover, we used enchant from the NLTK library to delete the typo word. By using enchant, it will be checked one by one word. When checking the words, there is no word in the English dictionary, the word will be deleted.

For example, this is an example processed data from Kaggle “A big thanks ds I got bst gd health”, by using a stopwords removal, noise character removal, and remove punctuation, the result is “big thanks ds got bst gd health” with remove “A” and “I”.

We tried to correct the words and the result is “big thanks ds i got BST gd health”. In this case, “bst” fixed by English word dictionary become “BST” (British Summer Time) because “BST” is in the English dictionary. The final step is to delete the typo words using enchant, the result is “big thanks i got health”. In enchant some words removed because there are not in the English dictionary.

C. Feature Extraction

For feature extracting the user’s review, the method provided by the NLTK toolkit was used.

1) Word2vec: Two main learning algorithm for Word2vec are skip-gram and bag-of-words. Bag-of-words will predict the word based on the content and the order of the words in history does not influence the projection. However, skip-gram will predict the surrounding words given the current word. The bag of words used a distributed representation of the context that different bag of words with skip gram. It is important to state that the weight matrix between the projection layer and input was shared for all the words positions. This paper used a modified Word2vec to calculate the document vector. For each word in the document will calculate the vector of a word and calculate the average of the document vector. This research used 300 dimensions for the word vector dimension size. The 300 vectors have calculated the mean for the number of words in the review.

1. $R(d_i) = \sum_t w2v(t)$ where $t \in d_i$
2. $w\_R(d_i) = \sum_t w_t\, w2v(t)$ where w_t = tf-idf weight of t
3. $C(d_i)$ = concatenate(tf-idf(di), $w\_R(d_i)$)

Fig. 2. Step Word2vec.

Utilization of word2vec and TF-IDF for feature classification is the same as [12]. In Fig. 2, d_i represents the document, the vector representation from Word2vec denoted as w2v(t), and t represents the term that exists in the document. The following step was done for each document. The first step is using Word2vec to summing the vector representation of a document, the second step is to calculate the TF-IDF value and the value applied in Word2vec, the last step is merging value of the TF-IDF and Word2vec which is weighted by TF-IDF from the second step [13].

2) TF-IDF: TF-IDF was used to specify the most common and used word in a corpus. TF-IDF used the word frequency to specify it. TF-IDF calculate the inverse proportion of the document to which the word appears in. More high the value of the TF-IDF, the more connection a word had for each other and the frequency of occurrence is also high. TF-IDF approach can be expressed as the equation below:

$$w_{t,d} = tf_{t,d}.log\ \frac{|D|}{|\{d\prime \in D | t \in d\prime\}|\prime} \quad (1)$$

where $tf_{t,d}$ is a term frequency of term t in document d, $log\ \frac{|D|}{|\{d\prime \in D | t \in d\prime\}|\prime}$ is inverse document frequency. |D| is the total number of documents in the documents set, $\{t \in d'\}$ is the number of documents containing the term t. When a word that repeatedly comes up is considered important words in the TF-IDF. As a result, the term TF-IDF is used to calculate the TF-IDF weights at the same time [10].

3) Doc2vec: For learning document embeddings, Doc2vec can be used as an extension to Word2vec. Dbow (Distributed Bag of Words version of Paragraph Vector) and dmpv (Distributed Memory version of Paragraph Vector) are two approaches of Doc2vec. Dbow and skip-gram works in the same way. The input in both approaches was replaced by a special token that represents the document. The words order in the document is ignored in this architecture. Dmpv has similarities with cbow. In the process, dmpv require an additional token document in addition to the word yet at this not conclude cbow but incorporating. The objective is again to predict a context word given the concatenated document and word vectors.

Fig. 3 shows two architectures from [18], the first is DMPV, and it will add a paragraph id to be trained with word vectors. This paragraph id contains information that is missing from the current word.

Fig. 4 shows ways to input into the DBOW model is a paragraph id, predicting randomly sampled words in this document [18].

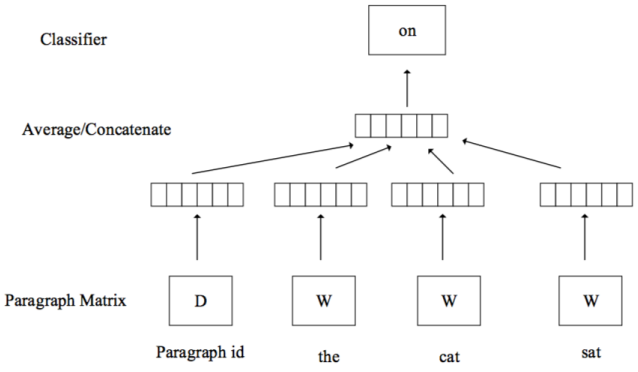

Fig. 3. DMPV

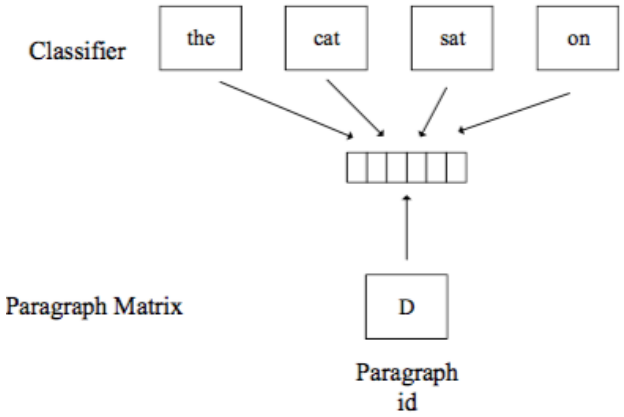

Fig. 4. DBOW

D. Classification

Using TF-IDF, Word2vec, and Doc2vec feature extraction, our initial approach was a summation of vectors in a particular document and then using linear Support Vector Machine (SVM), Naive Bayes, and Decision Tree to help classify them. It was implemented using the Sklearn library.

1) Support vector machine: Support Vector Machine (SVM) as a classifier is fully determined by a relatively small subset of the training instances, it is suitable for discrete data [18]. LinearSVC was used in this research as it is similar to SVM but with a kernel 'linear' parameter and implemented in terms of liblinear rather than libsvm. To analyze the complete vectorized data and the key idea behind the training of model that also known as SVM and to refine hyperplane. Equation represented by w was used [6].

$$\vec{w} = \sum_j a_j c_j \vec{d}_j, a_j \geq 0 \quad (2)$$

Dual optimization problem gives the value for a_j's. c_j (1,-1) be the class (positive, negative, neutral) for a document *dj* All the *dj* such that a_j is greater than zero are termed as support vectors as they are the only document vectors which are contributing to $\vec{w}$.

2) Naive bayes: This research using a Naive Bayes as a classifier. Bayes Theorem is the based properties of Naive

Bayes algorithm. This classifier works with assuming that each of every feature was standing as an independent individual. Though, this algorithm requires a small amount of data training that used for calculating the parameter for prediction and represented by P(c|d).

$$P(c \mid d) = \frac{P(d \mid c) * P(c)}{P(d)} \quad (3)$$

For a given textual review 'd' and for a class 'c' (positive, negative, neutral).

3) Decision Tree: Decision tree works with the three main components. The edges, internal nodes, and leaf nodes. Each of these components represents an item in text classification. The edges, internal nodes, and leaf nodes represent the test for feature weights, the feature, and categories resulted from the test. When the final node or the leaf nodes was reached, this represents the final category for the document. Decision tree had been used in many application in speech and language processing [1].

Classifier performance results will be generated as a confusion matrix. This matrix shows predictions of positive, negative, neutral predicted reviews. The number of correctly predicted negative reviews are called with True Negative, The False Negative Predicted Neutral is a false prediction that supposed to be a Negative but predicted as Neutral. False Negative Predicted Positive is a false prediction that a negative falsely predicted as positive. The False Neutral Predicted Negative and False Neutral Predicted Positive are the false prediction that supposed to be Neutral, but predicted as Negative and Positive. Last, The False Positive Predicted and False Positive Predicted Neutral is a false prediction that supposed to be positive but predicted as Negative and Neutral. The paragraph can be represented in Table II.

E. Parameter Optimization

Improvement or additional optimization is needed in the search strategy to get better performance when testing each new machine. This optimization still needed even though when the first run on a new machine gives a reasonable performance [20]. Hyperparameter optimization has strategies, they are grid search and manual search. Using this grid search will increase accurate by doing the checking of the parameters that are in the servant list, it will compare for the best accuracy. In this implementation, the results are not everything improves accuracy, there is some accuracy down.

TABLE II. CONFUSION MATRIX FOR CLASSIFIER

Correct Labels			
	Negative	**Neutral**	**Positive**
Negative	True Negative	False Negative Predicted Neutral	False Negative Predicted Positive
Neutral	False Neutral Predicted Negative	True Neutral	False Neutral Predicted Positive
Positive	False Positive Predicted Negative	False Positive Predicted Neutral	True Positive

Most widely used strategies for hyperparameter optimization are grid search and manual search [21]. Only one Hyperparameter that grid search can handle and the hyperparameter names and values have to be specified by the user [22].

The parameters used in this research are defined for the feature extraction parameters TF-IDF and BernoulliNB Classifier. feature extraction used a hyperparameter such as use_idf, and ngram_range for the vectorizer. For the classifier, this research used an alpha parameter for the BernoulliNB classifier. Many possible parameters can be used in this feature extraction and classifier such as 'dual', 'tol', 'C', and 'multi_class' for LinearSVC classifier; 'criterion', 'splitter', 'max_depth', and 'max_features' for the Decision Tree classifier; 'size', 'window' and 'min_count' for the Word2vec and Doc2vec feature extractions but limited by the computer resources.

This research used a different technique of hyperparameter optimization. For the Word2vec and Doc2vec, this research used an empty parameter. Only the TF-IDF and BernoulliNB that used a predefined hyperparameter as described in Fig. 5. Fig. 1 describes the experiment process about the hyperparameter optimization and the step that optimize with this optimization. The Train feature extraction and classification are those two that optimized with grid search.

The empty parameter might succeed to increase classification accuracy. This was possible because the model runs for the second time using the processed data. This might double trained the model and generate better and much higher accuracy for sentiment analysis classification.

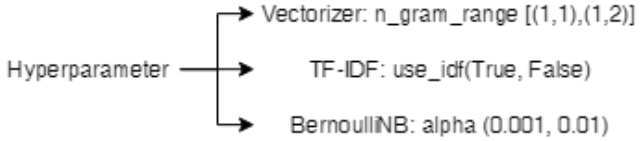

Fig. 5. Predefined Hyperparameter Example.

F. Evaluation

Evaluation in this research was done by calculating the classification accuracy score. The accuracy score calculates using The Sklearn library accuracy_score that calculate the number test data prediction that corrects divided by total testing data. T and F in the formula represent the True and False prediction. The Pos, Net, and Neg represent Positive, Neutral, and Negative.

$$Accuracy = \frac{TPos+TNet+TNeg}{TPos+FPos+TNet+FNet+TNeg+FNeg} \quad (4)$$

IV. RESULT AND DISCUSSION

This section will discuss the results and describe the limitations, threats to validity, and implication

A. Experiment Result

The experiment was done using 3 feature extractions as mentioned in Section 3. The TF-IDF generate the biggest feature size that contains all the vocabulary that exists the dataset and act as the columns. The Word2vec feature

extraction requires the most time to execute because it calculated each word vector and averages all the word vector in the same review to be the review/document vector.

In Table III, the experiment result shows that classification accuracy with data normal, data delete typo, and data spelling replaces that classified with LinearSVC, BernoulliNB, and Decision Tree classifiers. The classification in Table II was done using the default parameter. From the feature extraction point of view, TF-IDF produced the best accuracy for the default parameter with 81% average accuracy. From the classifier's point of view, LinearSVC has shown the best result with more than 84% in average accuracy much higher than Naïve Bayes and Decision Tree classifiers.

TF-IDF and LinearSVC can produce high accuracy rate is 85.66% in an average of 3 datasets. This result followed by the Decision Tree with TF-IDF feature extraction is 0.33% lower than TF-IDF while LinearSVC has resulted in 85.33% for the average of accuracy.

Table IV shows results from optimization with grid search optimization using the empty hyperparameter and predefined hyperparameter. The result has shown a better result for most feature extraction and classifier and changes from TF-IDF to Doc2vec as the best feature extraction accuracy with 85.33% on average. This result includes the predefined hyperparameter in Fig. 5 for the TF-IDF feature extraction and BernoulliNB classifier.

The result has shown that the predefined parameter succeeded in increased the accuracy for the TF-IDF and BernoulliNB classifier. In Table IV, accuracy for A-NB using normal data has succeeded increased by 1%. The result has shown a different result for Typo data and Spell data with 3% and 1% decreased each. The Bernoulli with TF-IDF also calculated using the empty hyperparameter and shows worse result than the predefined hyperparameter with 70%, 72%, and 70% for the normal, typo and spell data as present in Table V.

TABLE III. BEFORE OPTIMIZATION RESULT (A: TF-IDF, B: WORD2VEC, DOC2VEC)

Classification Results									
	Normal (%)			**Typo (%)**			**Spell (%)**		
	A	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
SVM	**86**	77	81	85	73	81	**86**	76	80
NB	73	63	75	76	58	77	74	62	79
DT	85	68	78	85	64	78	**86**	67	78

TABLE IV. AFTER OPTIMIZATION USING GRID SEARCH RESULT (A: TF-IDF, B: WORD2VEC, C: DOC2VEC)

Classification Results									
	Normal (%)			**Typo (%)**			**Spell (%)**		
	A	B	C	A	B	C	A	B	C
SVM	**89**	77	88	**89**	77	87	88	76	88
NB	74	62	84	73	60	82	73	60	85
DT	**89**	66	85	**89**	68	84	**89**	68	85

TABLE V. GRID SEARCH OPTIMIZATION DIFFERENCE RESULT USING PREDEFINED HYPERPARAMETER AND AN EMPTY PARAMETER

BernoulliNB with TF-IDF			
	Normal (%)	**Typo (%)**	**Spell (%)**
Empty Parameter	70	72	70
Predefined Parameter	74	73	73

Grid search hyperparameter is shown the best parameter with the best accuracy for the normal, typo and spell data. All normal, typo and spell data showed the same best parameter for grid search optimization with 'alpha': 0.01 , 'use_idf': True and 'ngram_range': (1,1). as mentioned before, not all data accuracy increased with this parameter, this possibly because of the small number of parameter that predefined for the grid search optimization. There might be another parameter combination that shown better result but can't achieve in this research cause of some limitations.

After optimization using empty parameter, the result has shown that TF-IDF and Decision Tree produces the best accuracy with 89% on average higher 0.33% from LinearSVC using the same TF-IDF feature extraction. The best feature extraction in this problem depends on the classifier that used. In average, the Doc2vec is the best on average for three classifiers. But for Decision Tree classifier, TF-IDF produces higher accuracy than Doc2vec.

Types of datasets we have though are divided into three parts, there are normal data, typo data, spell data. Normal data is the data that not preprocess using spell check and typo deletion, typo data is the review that preprocessed using the NLTK library to delete typo words, and spell data is the review data that pre-processed using the Autocorrect to correcting the misspelled words.

Fig. 6 represents the increasing accuracy rate of normal data using grid search hyperparameter optimization. Hyperparameter optimization succeeded to increase the classification accuracy by a combination of available parameters. The most promising result is Doc2vec with an 8.9% increase for the BernouliNB classifier.

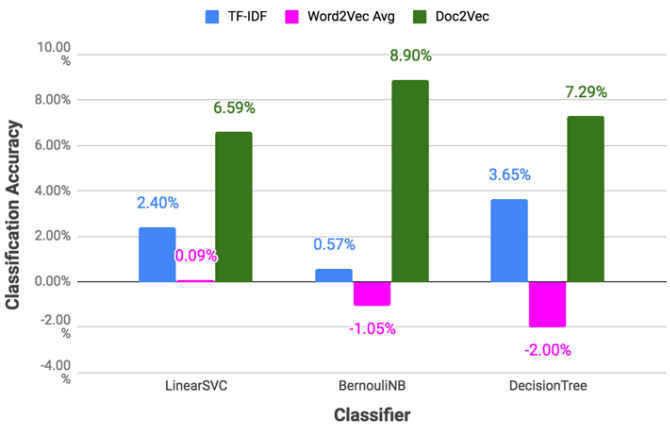

Fig. 6. Accuracy Chart using Normal Data.

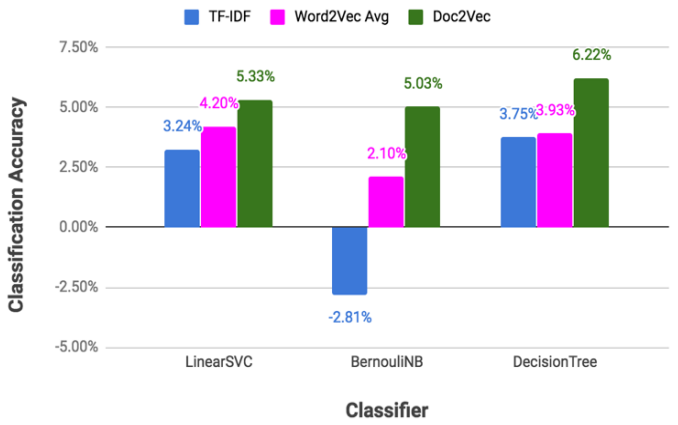

Fig. 7. Accuracy Chart using Typo Data.

Fig. 7 represents the increment accuracy rate using typo data. The chart showed all feature extraction and classifier has increased in a quite significant rate except for the Word2vec with BernoulliNB. The result showed that this method decreases 2.81% in accuracy from the original result.

Fig. 8 describes the increment accuracy rate with Spell data. In this chart, can be seen clearly that Doc2vec has increased the most. The summary for the optimization result is the Doc2vec has increased the most with 6.77% on average using normal, typo and spell data. Hyperparameter optimization using grid search manage to optimize the Doc2vec feature extraction for Google Play Review data.

B. Limitation

This research is limited by the computer resource to execute a calculation using bigger data and more complex optimization method. The computer used for this research has a relatively low spec with 16 RAM and just 2 CPU cores. These specs are low compared with [5] that supported with 2 Tesla K40 GPU. Other limitations for this research are the number of data available for google play apps review and available Python library. This research can be improved with better English spell replacer to fix the typo and much bigger dataset.

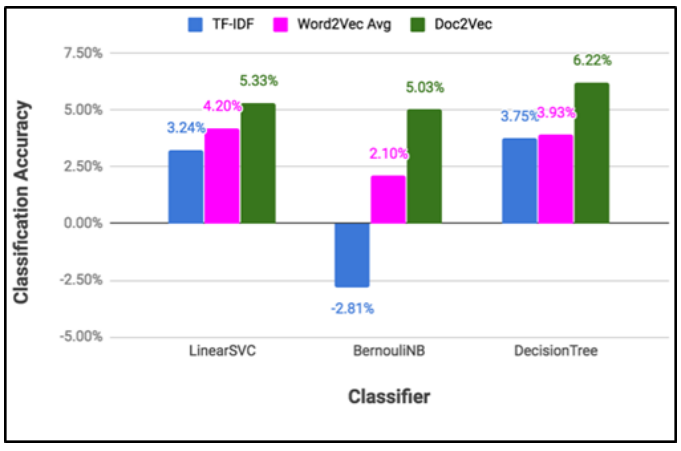

Fig. 8. Accuracy Chart using Spell Data.

C. Threats to Validity

The qualitative evaluation of the topic that relevant to the requirement engineering was done by the authors of the paper is one of the threats to validity. This is a threat as the evaluator could have an incomplete knowledge or misunderstanding about the specific information. Another threat to validity is the possibility of human error during the coding task. This human error can minimize with the second coder that double checking the code used for this research.

For text classification, many factors can influence the research. Data, method, variable, parameters and many more can make a different result. This result shows for normal data that included a typo and misspelled words, TF-IDF is the best feature extraction with SVM or Decision Tree Classifier. But, after optimizing the hyperparameter, the result shows that Doc2vec is the better result than others. The normal, typo and spell words do have a slight impact on the accuracy but did not have a significant influence. The typo and misspelled word only make around 2% in accuracy difference result.

V. Conclusion

This work presents a comparison between three feature extraction and a way to increase the classification accuracy for sentiment analysis. Before route optimization accuracy; TF-IDF using LinearSVC in normal data, TF-IDF using LinearSVC in spell data, and TF-IDF using DecisionTree in spell data have the same results is 86%. It can be concluded, in this study TF-IDF has the highest value. After hyperparameter optimization, the result has shown a different accuracy. After the optimization, there are accuracies up and down. TF-IDF using LinearSVC in normal data, TF-IDF using LinearSVC in spell data, TF-IDF using DecisionTree in spell data, and TF-IDF using DecisionTree in normal data have the same result i.e. 89%. Changes of accuracy made the Doc2vec to has the best accuracy results in total mean average. The increase in classification by hyperparameters optimization on the highest is Doc2vec using BernoulliNB in normal data increased by 8.9%.

Reference

[1] N. S. Joshi and S. A. Itkat, “A Survey on Feature Level Sentiment Analysis,” vol. 5, no. 4, pp. 5422–5425, 2014.

[2] P. Vateekul and T. Koomsubha, “A study of sentiment analysis using deep learning techniques on Thai Twitter data,” 2016 13th Int. Jt. Conf. Comput. Sci. Softw. Eng. JCSSE 2016, 2016.

[3] D. Zhang, H. Xu, Z. Su, and Y. Xu, “Chinese comments sentiment classification based on word2vec and SVMperf,” Expert Syst. Appl., vol. 42, no. 4, pp. 1857–1863, 2015.

[4] Y. Xi. Jin Gao, Yahao He, Xiaoyan Zhang, “Duplicate Short Text Detection Based on Word2vec,” pp. 3–7, 2017.

[5] X. Zhang, J. Zhao, and Y. Lecun, “Character-level Convolutional Networks for Text Classification,” pp. 1–9, 2015.

[6] A. Tripathy, A. Agrawal, and S. K. Rath, “Classification of Sentimental Reviews Using Machine Learning Techniques,” Procedia - Procedia Comput. Sci., vol. 57, pp. 821–829, 2015.

[7] J. H. Lau and T. Baldwin, “An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation,” 2014.

[8] E. R. Sparks et al., “Automating Model Search for Large Scale Machine Learning,” pp. 368–380.

[9] E. Guzman and W. Maalej, "How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews," vol. 200, Table V, pp. 86–87, 2014.

[10] R. Ju, P. Zhou, C. H. Li, and L. Liu, "An efficient method for Document categorization based on Word2vec and latent semantic analysis," Proc. - 15th IEEE Int. Conf. Comput. Inf. Technol. CIT 2015, 14th IEEE Int. Conf. Ubiquitous Comput. Commun. IUCC 2015, 13th IEEE Int. Conf. Dependable, Auton. Se, pp. 2276–2283, 2015.

[11] L. Lin, X. Linlong, J. Wenzhen, Z. Hong, and Y. Guocai, Text Classification Based on Word2vec and Convolutional Neural Network, vol. 3316. Springer International Publishing, 2018.

[12] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support Vector Machines and Word2vec for Text Classification with Semantic Features," 2015.

[13] D. Rahmawati and M. L. Khodra, "Word2vec semantic representation in multilabel classification for Indonesian news article," 4th IGNITE Conf. 2016 Int. Conf. Adv. Informatics Concepts, Theory Appl. ICAICTA 2016, pp. 0–5, 2016.

[14] S. K. R. Abinash Tripathy, Ankit Agrawal, "Classification of Sentiment Reviews using N-gram Machine Learning Approach," 2016.

[15] R. Ju, P. Zhou, C. H. Li, and L. Liu, "An Efficient Method for Document Categorization Based on Word2vec and Latent Semantic Analysis," 2015.

[16] W. Zhu, W. Zhang, G.-Z. Li, C. He, and L. Zhang, "A study of damp-heat syndrome classifcation Using Word2vec and TF-IDF," 2016.

[17] S. Fujita, "Text Similarity Function Based on Word Embeddings for Short Text Analysis," vol. 3406, pp. 391–402, 2018.

[18] Q. Shuai, Y. Huang, L. Jin, and L. Pang, "Sentiment Analysis on Chinese Hotel Reviews with Doc2Vec and Classifiers," 2018 IEEE 3rd Adv. Inf. Technol. Electron. Autom. Control Conf., no. Iaeac, pp. 1171–1174, 2018.

[19] S. R. Young, D. C. Rose, T. P. Karnowski, S. Lim, and R. M. Patton, "Optimizing Deep Learning Hyper-Parameters Through an Evolutionary Algorithm," 2008.

[20] J. Bilmes and J. Demmel, "Author Retrospective for Optimizing Matrix Multiply using PHiPAC : a Portable High-Performance ANSI C Coding Methodology," 2014.

[21] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," vol. 13, pp. 281–305, 2012.

[22] L. Kotthoff and K. Leyton-brown, "Auto-WEKA 2 . 0 : Automatic model selection and hyperparameter optimization in WEKA," vol. 18, pp. 1–5, 2017.