Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 6 No. 3 (2022) 352 - 358 ISSN Media Electronic: 2580-0760

The Accuracy Comparison Between Word2Vec and FastText On Sentiment Analysis of Hotel Reviews

Siti Khomsah[1], Rima Dias Ramadhani[2], Sena Wijayanto[3]
[1,2]Data Science, [3]Information Systems, Faculty of Informatics, Telkom Institute of Technology Purwokerto
siti@ittelkom-pwt.ac.id, rima@ittelkom-pwt.ac.id, sena@ittelkom.pwt.ac.id

Abstract

Word embedding vectorization is more efficient than Bag-of-Word in word vector size. Word embedding also overcomes the loss of information related to sentence context, word order, and semantic relationships between words in sentences. Several kinds of Word Embedding are often considered for sentiment analysis, such as Word2Vec and FastText. Fast Text works on N-Gram, while Word2Vec is based on the word. This research aims to compare the accuracy of the sentiment analysis model using Word2Vec and FastText. Both models are tested in the sentiment analysis of Indonesian hotel reviews using the dataset from TripAdvisor.Word2Vec and FastText use the Skip-gram model. Both methods use the same parameters: number of features, minimum word count, number of parallel threads, and the context window size. Those vectorizers are combined by ensemble learning: Random Forest, Extra Tree, and AdaBoost. The Decision Tree is used as a baseline for measuring the performance of both models. The results showed that both FastText and Word2Vec well-to-do increase accuracy on Random Forest and Extra Tree. FastText reached higher accuracy than Word2Vec when using Extra Tree and Random Forest as classifiers. FastText leverage accuracy 8% (baseline: Decision Tree 85%), it is proofed by the accuracy of 93%, with 100 estimators.

Keywords: *word2vec, fast text, sentiment analysis, hotel review*

1. Introduction

The disadvantage of Bag-of-Word (BoW) vectorization is the loss of information related to the context of sentences, word sequences, and semantic relationships between words in sentences. The BoW method only notices a word as a standalone object. Another drawback is storage issues. When vectorization is applied to a large corpus, the BoW requires a large matrix space. All vocabulary in the corpus is multiplied by the number of sentences. In the large matrix, most of the value is zero number[1]. This weakness was solved when in 2000, Word embedding was discovered by Mikolov[2]. Word embedding maps each word in a sentence into a vector by paying attention to the closest's word. The purpose of mapping is to gain the semantics of the word. Mikolov also introduced Word2Vec with its two models: Skip-gram and Continuous Bag-of-Words (CBoW) [3]. Then in 2017, Facebook introduced the FastText model that used sub-word information[4].

Several studies have stated that word2Vec's accuracy is higher than FastText, especially in text analysis in Indonesian. Saputri's proposed [5] emotion analysis on opinion datasets from Twitter using word2vec and FastText, is combined with Logistic Regression [5]. The F1-Score of Word2Vec is 67.32%, while FastText is 66.46%. The research of Sazany and Budi [6] concludes that FastText is better than Word2Vec for text analysis in identifying hate speech on Indonesian text data from Twitter opinions. The proposed method in [6] provides the best result in F1-Score on FastText rather than Word2Vec using the LSTM (Long Short-Term Memory) model as a classifier. In other research, Hasanah [7] has a conclusion that FastText reaches higher accuracy than Word2Vec. Hasanah [7] classified the COVID-19 tweets dataset into seven classes: warnings and suggestions, information notification, donations, emotional support, seeking help, criticism, and hoaxes. The classification was tested using a combination of word embedding (Word2Vec and FastText) with deep learning methods (CNN, RNN, and LSTM). The results showed that the highest accuracy was 97.3% and 99.4% when using a combination of FastText and LSTM [7]. In other research, Riza and Charibaldi [8] tested Word2Vec and FastText on emotion detection on Twitter using LSTM. The results

Accepted: 15-12-2021 | Received in revised: 15-03-2022 | Published: 30-06-2022

showed that the FastText and Word2Vec methods gave the same accuracy of 73.15% [8].

Those studies above show classification accuracy will reach above 90% if word embedding is combined with deep learning. In previous research, combining word embedding with some ensemble machine learning is rare. Therefore, research on word embedding (Word2Vec or FastText) still needs to be continued for datasets and other machine learning methods. So, our research aims to compare the performance of FastText and Word2Vec. We use four machine learning, namely Decision Tree, Random Forest, Extra Tree, and AdaBoost.

2. Research Methods

The first step is processing unstructured text into semi-structured. The semi-structured text was converted into vectors using Word2Vec and FastText. Opinion vectors are classified using four machine learning algorithms, namely Decision-Tree (DT), Random Forest (RF), Extra Tree (ET), and AdaBoost (AB). Specifically for ensemble methods, the study used 100 estimators (trees) on Random Forest (RF), Extra Tree (ET), and AdaBoost (AB) models. The dataset is split into data training of 80% and data testing of 20%. The steps of the process of sentiment analysis shown in Figure 1.

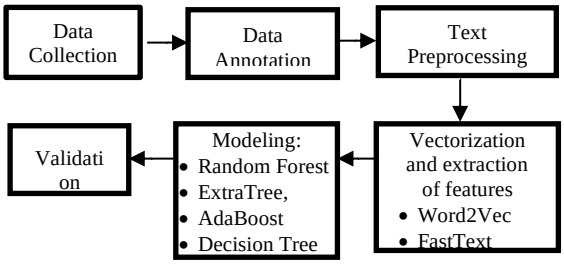

Figure 1. Research Process Flow

2.1 Data Collection

Our experiment used the hotel reviews in Purwokerto City, Central Java, Indonesia. Datasets were downloaded from TripAdvisor. These reviews were accompanied by ratings given by visitors in the form of ratings in five categories: extraordinary, very good, average, bad, and very bad. The number of datasets collected is 3145 comments.

2.2 Data Annotation

Data Annotation is the process of determining the class/label of each comment (review). The label of each review is determined based on the rating by the visitor. The sentiment labels in this study are positive and negative, as shown in Table 1.

Table 1. Label Class

Label	Description	Symbol
Positive	Positive Sentiment	1
Negative	Negative sentiment	-1

Visitor ratings are five levels, indicated in Table 2. The rating needs to be converted into positive and negative sentiments. The ratings of "Extraordinary" and "Excellent" were changed to positive labels (1). The ratings of "Average", "Bad", and "Very Bad" are changed to negative labels (-1). The "average" rating is included in the negative label. Based on observations, we found many negative keywords in reviews with an "average" rating. Examples of labeling results are in Table 2.

Table 2. Examples of Opinions from Datasets and Annotation Conversions

No	Reviews	Rating from User	Convert to Sentiment
1.	Hotel Y ini memberikan pelayanan terbaik yang tidak pernah saya dapatkan di hotel manapun. Hotel Y benar-benar telah memberikan pelayanan berkelas pada tamunya. Propertinya terawat dengan sangat baik dan sangat bersih. Makanannya luar biasa. Staffnya sangat membantu dan ramah. Fasilitasnya sangat sangat bagus.	Extraordinary	1
2.	Lokasi yang strategis di puncak bukit dengan ruangan dan lobi yang bagus. Tapi hotel ini punya beberapa kekurangan. Sambungan internet tidak bekerja baik di kamar ataupun lobi. Air panas jarang tersedia dan sarapan paginya hanya minuman jus, roti panggang, teh dan 2 telur siap-masak untuk dibeli.	Average	-1
3.	Tinggal di sini satu malam di bulan Juli. Mengerikan hotel: tidak ada wiski di bar, atau cappuccino, tidak ada makanan yang baik, tidak ada air di kolam renang, kecuali kami dan pasangan lain tidak ada tamu - saya bertanya-tanya mengapa. . . ;) Hotel terburuk selama perjalanan kami melalui indah Jawa!	Very Bad	-1

Based on the result of data analysis, all comments on TripAdvisor are not single sentences. Most comments are paragraphs consisting of several sentences. Positive and negative comments can be found in one opinion. The example of sentence number one in Table 2 shows an opinion with only positive polarity. In the second opinion, the word "good" contains positive elements, while the elements of the word meaning negative are the word "not working" and the word "lack". Reviews consisting of several sentences or paragraphs, such as number 3 in Table 2 are compound sentiments. In the reviews, the negative elements look more like the words "terrible" and "worst". Then the number three review with a very bad rating is grouped as a negative opinion. The results of the labeling process obtained a dataset with a positive label of 2425 and a negative label of 720 comments.

DOI: https://doi.org/10.29207/resti.v6i3.3711

2.3 Text Preprocessing

Hotel service user reviews on TripAdvisor are unstructured. To convert into a semi-structured form, it needs a preprocessing stage. A detailed description of the preprocessing stage is as follows:
(1) Remove non-alphabetic symbols and numbers. If a number, character, or non-alphabetic symbol such as the sign *, & #, @, (, [,], $ is found, then the symbol will be removed.
(2) Delete the stop-word. Stop-words are words that do not contain elements of sentiment. It will be deleted if it is found in the stop words list. List of stop-words obtained from previous research [9].
(3) Removes recurring characters into a single character. For example, the word "heeebatt" will change into "hebat". "Ngeeriii" will convert into "ngeri", and "maantaapp" into "mantab".
(4) Remove words that are a single character, such as "y", "t", and so on, that are commonly used to symbolize the word "ya(yes)", "tidak(no)".
(5) Convert the slang word into the formal Indonesian (KBBI). For example, "bngeet" will convert into "banget", "eloo" into "kamu", "pucing" into "pusing", and so on. To support the conversion, we use a slang word dictionary. The dictionary structure of slang words is in Table 3.

Table 3. Slang Word Dictionary Structure

Slang Word	Formal Word
…	…
Bnget	Banget
Elo	Kamu
Gue	Saya
…	…

The process of converting slang words into raw words is in the following pseudocode.

```
def konversi_slangword(kalimat,kamus)
kalimatbaru←""
if length(kalimat)>0
  for kata in kalimat.split()
     katabaru←kata
     for i ← 1 to length(kamus)
         if katabaru==kamus[i]
            katabaru←kamus[i]
           exit for
         endif
     next i
     kalimatbaru←kalimatbaru+katabaru
  next kata
end if
return kalimatbaru

kamus=[['kataslang1','kataformal1,['kataslang2
      ','kataformal2'],..]
kalimat=’opini mengandung slank word'
hasil=def_konversi_slangword(kalimat,kamus)
```

The slang words conversion algorithm in the pseudocode above works for each sentence. Each sentence in the dataset will be sent to the slang words conversion function. Sent sentences are subject to a tokenization process (separation for each word). Each word will be searched in the slang word dictionary. If found, it will be replaced with a common word. New sentences without slang will be re-assembled as the pseudocode output.

2.4 Vectorization and Extraction Features

The sentiment classification model in the study used Word2Vec and FastText for vectorization and extraction features. Both methods are given the same parameter values: the number of features or word vector dimensionality is determined at 100, the minimum word count value is 2, the number of parallel threads is 4, and the size of the context window is 5.

1) Vectorization: Word2Vec

Word2Vec result vector matrix is smaller than the BoW vector matrix. The size of the vector matrix depends on the number of features and the number of words. There are two Word2Vec architectures: Continuous Bag-of-Words (CBoW) and Skip-gram [10]. The study used the Skip-gram model. The way it works is to consider the proximity of the words around (or referred to as the target word) a primary word (or referred to as the context word). To determine the target word required window size parameters, namely the number of words that will be checked before and after the context word [11][12]. Figure 2 shows an example of a Skip-gram with a window size of 2.

For example, an opinion: "Hotel Melati paling bersih" (the cleanest Melati hotel). If the context word is the word "hotel", then the target word on the left does not exist (empty), and on the right is the word "Melati" and the word "paling". Then, the pair of context word and target word produced is {("hotel", "Melati"), ("hotel", "paling")}[9]. In Figure 2 the context word pair and the word target are inputs from the Skip-gram architecture.

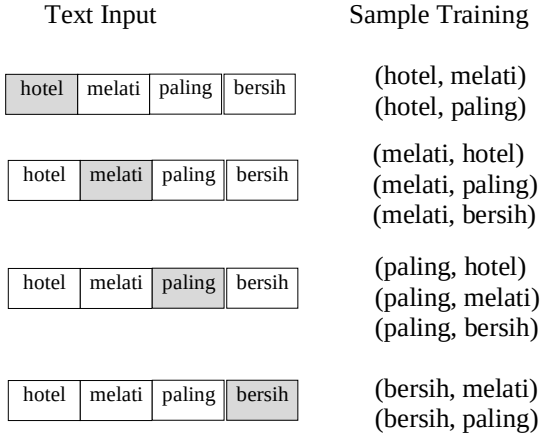

Figure 2. Skip-Gram Window Illustration [9]

DOI: https://doi.org/10.29207/resti.v6i3.3711

2) Vectorization: FastText

FastText developed from Word2Vec. FastText's vectorization process is more detailed than Word2vec. FastText's word representation approach is different from Word2Vec. Word2Vec uses each word as the smallest unit, while FastText uses the syllable (N-Gram) as the smallest unit. The arrangement of syllables in FastText uses the N-Gram principle. The length of N will be formed according to the length of the word[13]. Therefore, the vector is formed from that number of N-Gram. That principle causes the need for additional processes at each stage of the training data[14]. For example, the word "Selamat" will be encoded based on the sum of the N-Gram vector:

<se,<sel,<sela,<selam,sel,sela,selam,sel
ama,ela,elam,elama,elamat,lam,lama,lamat,la
mat>,ama,amat,amat>,mat,mat>,at>

The < and > signs indicate the beginning and end of the word. FastText vectors prove to be more accurate than Word2Vec but with much larger vector sizes. Thus, the FastText process is certainly more detailed but will be longer than Word2Vec. However, the N-Gram feature on FastText is the most significant improvement of Word2Vec. This improvement is designed to solve the "out of vocabulary" error. When word-embedder searches for the word "selametan" in a Word2Vec vector, it may not find the corresponding word. But FastText can guess with various passages in the word "Selamat" and the word "selametan", to state that "selametan" is close to the word "Selamat". The specialty of FastText vectorization needs to be seen in its performance in this study.

Both word embedding methods will be tested with 4 (four) ensemble machine learning methods: Decision Tree, Random Forest, Extra Tree, and Adaboost. The Decision tree (DT) method is used as a baseline because this method is the basis of the Random Forest, Extra Tree, and AdaBoost methods.

2.5 Modeling

Ensemble machine learning methods such as Random Forest and Extra Tree are known to be effective for classifying data imbalance [15]. Datasets obtained from TripAdvisor are imbalanced. The number of datasets labeled positive sentiment is more than negative. Therefore, this study is proper when using modeling algorithms to overcome these conditions. Machine learning to be used in this study are Decision Tree, Random Forest, Extra Tree, and AdaBoost.

1) Decision Tree

The Decision Tree is an algorithm that structures based on features in datasets. Several ways can be used to arrange the Tree's structure, namely CART, ID3, and C4.5. C4.5 algorithm was developed from ID3 algorithm [16]. The C4.5 algorithm builds a Decision Tree by selecting the primary attribute as the root point. It is followed by forming branches, then dividing data into each of these branches. While the branches of the Tree are formed using the value of entropy [17]. The process is repeated until all data is positioned in the right branch.

The shape of the Decision Tree model for the classification of datasets according to the research case is in Figure 3. The branching point on the Decision Tree is a feature of the word that has a great entropy or Gini index value. The arrow-shaped branch line indicates a class that is positive or negative. The line contains a threshold value. If data is more than the threshold, it is then classified into 0 (negative). Vice versa, it goes to the point of the next branch, or the leaf is worth 1. A leaf or class of 0 is a negative sentiment, and a leaf or class worth 1 is a positive sentiment.

2) Random Forest

Random Forest is quite popular in sentiment analysis [8]-[12]. Random Forest is a development of the Decision Tree. Random Forest applies the ensemble principle: to build many Decision Trees from several sub-datasets. The sub-dataset is called bootstrap. Splitting the dataset into sub-dataset is to reduce the risk of overfitting. Random Forest separates the nodes in each Decision Tree in the best split. An example of the process of bootstrap formation and construction of several trees in the random forest is in Figure 4. Each Decision Tree will give the results of its classification. The decision of the classification results is obtained by voting from the results of the classification of each Decision Tree (majority class).

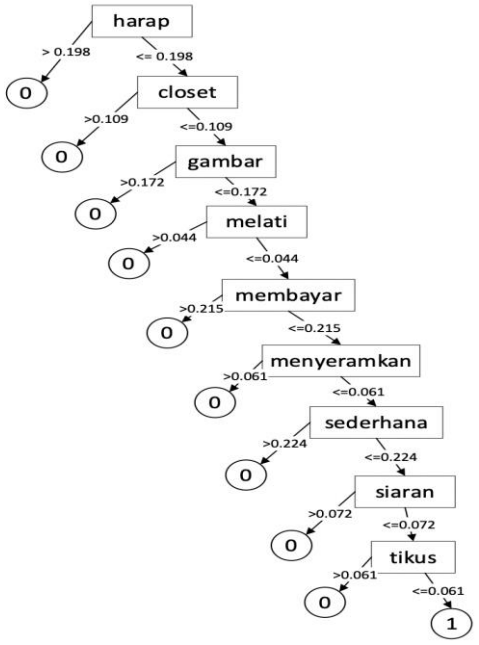

Figure 3. Decision Tree Structure

DOI: https://doi.org/10.29207/resti.v6i3.3711

3) Extra Tree

Extra Tree is an ensemble method that works well for sentiment analysis outperforming basic algorithms such as KNN, Naive Bayes, and Decision Tree [18]. Extra Trees are also capable of working on unbalanced data [19]. As with the Random Forest method, Extra Trees is an ensemble form of the basic Decision Tree algorithm. The Extra Tree algorithm works similarly to Random Forest by creating multiple bootstraps and generating Decision Trees based on those bootstraps. Classification decisions are determined based on the majority of decisions from the Decision Tree [20]. For example, the majority of the prediction class of each Decision Tree is "Class X" then the final decision is "Class X". The difference between Extra Tree and Random Forest is in bootstrapping. Random Forest forms bootstrap by randomly retrieving data from datasets.

In contrast, Extra Tree does not take data randomly. Another difference is in the construction of each Decision Tree. Extra Tree applies random-split to define nodes in Decision Tree, while Random Forest applies best-split.

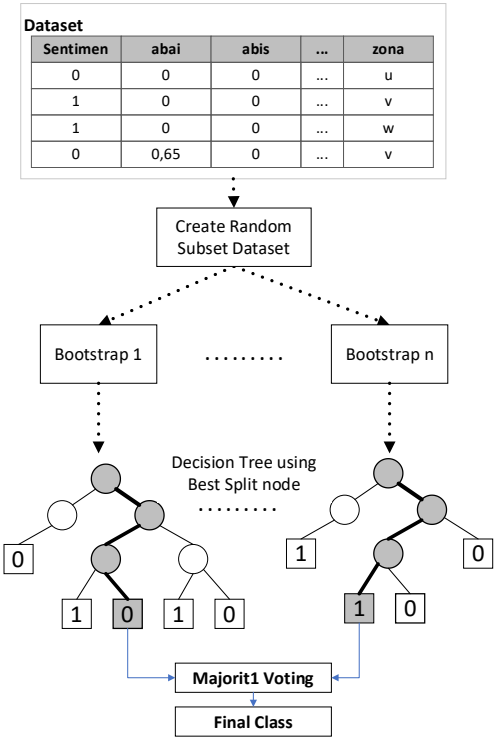

Figure 4. Random Forest

4) AdaBoost

AdaBoost has a higher classification performance in some studies than Random Forest [21]. An AdaBoost is one ensemble machine learning that uses the boosting principle. In the Boosting algorithm, each classifier is trained on the data, using the success of the previous classifier. Each step of the training is completed, and the weight of the data is re-distributed. Data that is misclassified will be increased in weight to indicate that the data is difficult to classify. In this way, the following learning process will calculate the loss function in the form of exponential functions to improve accuracy [22].

2.6 Validation

In the machine learning classification model, validation is required for evaluation. The performance of the model is measured by the confusion matrix that was also used in the study. The confusion matrix contains actual numbers and predicted results numbers. The measures in the matrix confusion are accuracy, precision, recall, and F1-Score values.

3. Result and Discussion

The study's results were model validation measurements based on four standard performance measures: accuracy, precision, recall, and F1-Score. The results of the study are in Table 4. The Decision Tree (DT) method is used as a baseline.

Table 4. Research Results

Vectorizer	Machine Learning	Result				
		Accuracy	Sentiment	Precision	Recall	F1-Score
Word2Vec	DT	0,86	+	0.90	0.92	0.91
			-	**0.72**	**0.68**	**0.70**
	RF	**0.92**	+	0.96	0.94	0.95
			-	0.78	0.86	0.81
	ET	**0.92**	+	0.97	0.93	0.95
			-	0.75	0.87	0.80
	AB	0.82	+	0.91	0.86	0.89
			-	0.47	0.60	0.53
Fast-Text	DT	0.85	+	0.88	0.92	0.90
			-	0.75	0.63	0.68
	RF	**0.93**	+	**0.97**	**0.94**	**0.95**
			-	0.78	0.87	0.82
	ET	**0.93**	+	**0.97**	**0.94**	**0.96**
			-	**0.80**	**0.88**	**0.84**
	AB	0.82	+	0.91	0.86	0.89
			-	0.47	0.60	0.53

Note: RF = Random Forest; ET= Extra Tree; AB= AdaBoost; DT= Decision Tree.

We use Decision Tree as a baseline for comparing the impact of FastText and Word2Vec toward ensemble classifiers (Random Forest, Extra Tree, and AdaBoost). In Table 4, Decision Tree- Word2Vec has an accuracy of 86%, while Decision Tree- FastText has 85%. On ensemble, the results show that the best classifier model is FastText-Extra Tree and FastText-Random Forest, which reach an accuracy of 93%, which is higher than other models. F1-Score is 97% for Extra Tree on positive sentiment and 84% on negative sentiment. In positive sentiment, the precision of Extra Tree is 97%, and recall is 94%, while in negative sentiment, their precision and recalls are 80% and 88%, respectively. F1-score of Random Forest is 95% on positive sentiment and 82% on negative sentiment. Meanwhile, precision and recall of Random Forest are 78% and 87%, respectively,

DOI: https://doi.org/10.29207/resti.v6i3.3711

slightly higher than Extra Tree. From these results, we can conclude that these ensemble model using either Word2Vec or FastText only works well on positive sentiment datasets. Imbalanced datasets probably cause it. Imbalanced shown by positive data is more than negative, and it could be a concern in the following research to develop consistent models on balance datasets.

According to the negative dataset sentiment in Table 4, unexpectedly, the Decision Tree model is more accurate than AdaBoost when using Word2Vec or FastText. Although we know that AdaBoost works boosting methods to increase accuracy, when combined with Fast Text and Word2Vec, it can't increase accuracy. Decision Tree reaches an accuracy of 85%- 86%, higher than AdaBoost with 82%. However, the accuracy performance of the DT-Word2Vec or DT-Fast Text models only works well for the positive sentiment dataset. Meanwhile, in the negative sentiment dataset, the model is still inconsistent, as indicated by their precision and recall, which differ significantly from the accuracy. In general, FastText provides higher accuracy for ensemble models.

4. Conclusion

The results of the study showed that the best vectorizer is FastText. The best model is the combination of FastText and Extra Tree. Bagging on Random Forest and Extra Tree accounted for high accuracy rather than the boosting on AdaBoost. Our recommendation for further research is to combine ensemble machine learning with other vectorizers, such as Glove or Wang2Vec, and also try several estimators, different parameters, and various enormous datasets.

Acknowledgment

The authors thank LPPM Telkom Institute of Technology Purwokerto for funding this research.

References

[1] A. Nurdin, B. A. S. Aji, A. Bustamin, and Z. Abidin, "Perbandingan Kinerja Word Embedding Word2Vec , Glove ," *Jurnal TEKNOKOMPAK*, vol. 14, no. 2, pp. 74–79, 2020.

[2] P. Linguistics, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017, [Online]. Available: https://transacl.org/ojs/index.php/tacl/article/view/999.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv:1301.3781 [cs]. Available at: http://arxiv.org/abs/1301.3781*, no. January 2013, 2014.

[4] S. Thavareesan and S. Mahesan, "Sentiment Lexicon Expansion using Word2vec and fastText for Sentiment Prediction in Tamil texts," *Mercon 2020 - 6th International Multidisciplinary Moratuwa Engineering Research Conference, Proceedings*, pp. 272–276, 2020, DOI: 10.1109/MERCon50084.2020.9185369.

[5] M. S. Saputri, R. Mahendra, and M. Adriani, "Emotion Classification on Indonesian Twitter Dataset," *Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018*, pp. 90–95, 2019, DOI: 10.1109/IALP.2018.8629262.

[6] E. Sazany and I. Budi, "Deep Learning-Based Implementation of Hate Speech Identification on Texts in Indonesian: Preliminary Study," *Proceedings of ICAITI 2018 - 1st International Conference on Applied Information Technology and Innovation: Toward A New Paradigm for the Design of Assistive Technology in Smart Home Care*, pp. 114–117, 2018, DOI: 10.1109/ICAITI.2018.8686725.

[7] N. A. Hasanah, N. Suciati, and D. Purwitasari, "Identifying Degree-of-Concern on COVID-19 topics with text classification of Twitters," *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, vol. 7, no. 1, pp. 50–62, 2021, DOI: 10.26594/register.v7i1.2234.

[8] M. A. Riza and N. Charibaldi, "Emotion Detection in Twitter Social Media Using Long Short-Term Memory (LSTM) and Fast Text," *International Journal of Artificial Intelligence & Robotics (IJAIR)*, vol. 3, no. 1, pp. 15–26, 2021, DOI:10.25139/ijair.v3i1.3827.

[9] S. Khomsah, "Sentiment Analysis On YouTube Comments Using Word2Vec and Random Forest," *Telematika*, vol. 18, no. 1, p. 61, 2021, doi: 10.31315/telematika.v18i1.4493.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pp. 1–12, 2013.

[11] X. Rong, "word2vec Parameter Learning Explained," pp. 1–21, 2014.

[12] C. Mccormick, "Word2Vec Tutorial - The Skip-Gram Model," 2016.

[13] B. Kuyumcu, C. Aksakalli, and S. Delil, "An automated new approach in fast text classification (FastText): A case study for Turkish text classification without preprocessing," in *ACM International Conference Proceeding Series*, 2019, pp. 1–4, DOI: 10.1145/3342827.3342828.

[14] S. Tiun, U. A. Mokhtar, S. H. Bakar, and S. Saad, "Classification of functional and non-functional requirement in software requirement using Word2vec and fast Text," *Journal of Physics: Conference Series*, vol. 1529, no. 4, 2020, DOI: 10.1088/1742-6596/1529/4/042077.

[15] A. S. More and D. P. Rana, "Review of random forest classification techniques to resolve data imbalance," *Proceedings - 1st International Conference on Intelligent Systems and Information Management, ICISIM 2017*, vol. 2017-Janua, pp. 72–78, 2017, DOI: 10.1109/ICISIM.2017.8122151.

[16] M. Akhtar and R. S. Parihar, "An Hybrid Data Mining Approach to detection and classification of Health Care Data," *International Journal of Electrical, Electronics and Computer Engineering*, 2017.

[17] A. K. Mohamad, M. Jayakrishnan, and N. H. Nawi, "Employ Twitter Data to Perform Sentiment Analysis in the Malay Language," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 2, pp. 1404–1412, 2020, DOI: 10.30534/ijatcse/2020/76922020.

[18] A. S. Aribowo, H. Basiron, N. F. A. Yusof, and S. Khomsah, "Cross-Domain Sentiment Analysis Model On Indonesian Youtube Comment," *International Journal of Advances in Intelligent Informatics*, vol. 7, no. 1, pp. 12–25, 2021, DOI: 10.26555/ijain.v7i1.554.

[19] D. Tiwari and N. Singh, "Ensemble Approach for Twitter Sentiment Analysis," *I.J. Information Technology and Computer Science*, vol. 8, no. August, pp. 20–26, 2019, DOI: 10.5815/ijitcs.2019.08.03.

[20] A. S. Aribowo, H. Basiron, N. S. Herman, and S. Khomsah, "An Evaluation of Preprocessing Steps and Tree-based Ensemble Machine Learning for Analysing Sentiment on Indonesian YouTube Comments," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 5, pp. 7078–7086, 2020, DOI: 10.30534/ijatcse/2020/29952020.

[21] D. Ganatra and D. Nilkant, "Ensemble methods to improve accuracy of a classifier," *International Journal of Advanced*

Trends in Computer Science and Engineering, vol. 9, no. 3, pp. 3434–3439, 2020, DOI: 10.30534/ijatcse/2020/145932020.

[22] Zulhanif, "Algoritma AdaBoost Dalam Pengklasifikasian," in *Prosiding Seminar Nasional Matematika dan Pendidikan Matematika UMS 2015*, 2015, pp. 559–569, doi: 10.1017/cbo9781139028462.008.

DOI: https://doi.org/10.29207/resti.v6i3.3711