JURNAL INFOTEL
VOL. 17, NO. 1, FEBRUARY 2025, PP. 54–67. DOI:10.20895/INFOTEL.V17I1.1077

RESEARCH ARTICLE

The Evaluation of Effects of Oversampling and Word Embedding on Sentiment Analysis

Nur Heri Cahyana[1], Yuli Fauziah[2], Wisnalmawati[3], Agus Sasmito Aribowo[4,*] and Shoffan Saifullah[5]

[1,2,4]Department of Informatics, Universitas Pembangunan Nasional "Veteran", Yogyakarta 55283, Indonesia
[3]Faculty of Economics and Business, Universitas Pembangunan Nasional "Veteran", Yogyakarta 55283, Indonesia
[4]Fakulti Teknologi Maklumat dan Komunikasi, UTeM, Melaka 76100, Malaysia
[5]AGH University of Krakow, Krakow 30-065, Poland

*Corresponding email: sasmito.skom@upnyk.ac.id

Received: November 2, 2023; Revised: February 14, 2025; Accepted: February 27, 2025.

Abstract: Unbalanced datasets in sentiment analysis present a persistent challenge, such as often exhibit a bias favoring the majority class in classification. To address this issue, researchers employ oversampling to rectify the imbalance by introducing synthetic data. However, this approach leads to larger datasets that demand more time and resources for model training, also potentially resulting in overfitting. This research aims to balance the data through oversampling while using static word embedding techniques (Word2Vec and FastText). The research process begins by converting opinion data into sentence vectors using Word2Vec and FastText, followed by oversampling methods. Five datasets opinion varying in record count and imbalance levels, are used for experimentation. The study demonstrates improvements in sentiment analysis accuracy when Word2Vec or FastText is combined with three oversampling techniques: SMOTE, Borderline SMOTE, or ADASYN. To mitigate overfitting, Random Forest is used in the classification models. Performance assessment is based on accuracy and the F-measure. Following extensive testing, it is observed that the performance of the Word2Vec method aligns closely with that of FastText. The borderline SMOTE emerges as the most effective oversampling method. Combining Word2Vec or FastText with Borderline SMOTE is shown to be the optimal choice, providing accuracy and scores of the F-measure in the range of 91.0% to 91.3%. Sentiment analysis models utilizing Word2Vec or FastText in conjunction with Borderline SMOTE exhibit the potential for high-performance alternatives.

Keywords: oversampling, sentiment analysis, word embedding

1 Introduction

Sentiment analysis is a critical aspect of Natural Language Processing (NLP) that has garnered significant research interest. Studies in this domain focus mainly on the development of efficient models to achieve high performance [1]. However, supervised sentiment analysis relies on annotated datasets, which often exhibit class imbalance. In such cases, the majority class (*e.g.*, positive sentiment) significantly outweighs the minority class (*e.g.*, negative sentiment), leading to classification bias [2]. To address this issue, data balancing techniques such as under-sampling and oversampling can be applied. Oversampling techniques, such as the Synthetic Minority Over-sampling Technique (SMOTE), generate synthetic instances for the minority class, preserving original data and enhancing model performance. However, a key challenge remains to determine the optimal synthetic data to generate.

Previous research has explored the combination of oversampling and machine learning for sentiment analysis. A study implemented Random Forest with SMOTE, achieving a precision of 71%, a recall of 70%, and a precision of 70% [3]. Another study used Latent Dirichlet Allocation (LDA) and Support Vector Machine (SVM) with SMOTE-Tomek Links, achieving 98% precision, precision, recall, and F1-score [4]. However, these studies were conducted on small datasets, limiting their generalizability. Similar research applied LSTM with SMOTE, obtaining an accuracy of 89.42% on a dataset of only 1,903 samples [5], another study that employed multiple machine learning models together with SMOTE reported an accuracy of 91% on a dataset consisting of 1,770 samples [6].

Despite promising results, several limitations remain. Some approaches, such as Random Forest with SMOTE, yield suboptimal accuracy, while others, such as SVM with SMOTE-Tomek Links, perform well but lack scalability due to small dataset size. Furthermore, previous studies have typically relied on a single oversampling technique or word embedding method, potentially limiting the robustness of the model. These gaps highlight the need for further research that integrates multiple oversampling strategies with word embedding techniques, tested on larger and more diverse datasets.

Previous studies have shown that oversampling improves classification quality, with Borderline-SMOTE achieving the highest accuracy [7]. However, previous research has not extensively examined the combination of oversampling techniques with word embedding, particularly for sentiment analysis in the Indonesian language. Furthermore, another study explored sentiment analysis using word embeddings derived from Stack Overflow (SO) posts and Google News. This research further examined the impact of two machine learning techniques oversampling and undersampling on dataset balancing. The results demonstrated that oversampling effectively improved the performance of the sentiment classifier [8]. Based on [7–9], it can be concluded that SMOTE is the most widely used oversampling method. However, no research has examined the effectiveness of these oversampling techniques when combined with Word2Vec and FastText, particularly for sentiment analysis in the Indonesian language. Furthermore, no studies have investigated the application of Borderline SMOTE and ADASYN in similar cases.

This study evaluates three widely used SMOTE variations: SMOTE, Borderline-SMOTE, and ADASYN are investigated.

1.1 SMOTE

SMOTE refers to an oversampling technique resulting in a set of synthetic data by making a new instance between two features of two data present in one class. This synthetic data is obtained through the following steps.

(a) Taking feature x from the dataset in the minority class and finding k closest to feature y in a similar minority class.

(b) For each chosen feature y, calculate the distance difference between feature x and feature y, then multiply the distance difference with a random number between 0 and 1.

(c) Then, add the feature y to the part of the minority class [9].

1.2 Borderline-SMOTE

The advanced method of SMOTE is Borderline-SMOTE, which is believed to have a better performance. Its mechanism starts by seeking members of the minority class in the borderlines. Then it is continued by making synthetic data from the borderlines. The synthetic data are then united with a set of original data in a minority class until a balance is reached [9].

1.3 ADASYN

ADASYN is a method for the oversampling approach using the distribution weight for the data in the minority class based on the difficulty level of learning. Synthetic data are the result of the minority class that is difficult to learn rather than the minority class that is easy to learn [9].

Unlike random oversampling, which duplicates minority class samples, these techniques create synthetic data, reducing the risk of overfitting. This study investigates whether combining oversampling techniques with static word embedding methods, such as Word2Vec and FastText, can enhance sentiment analysis performance. The process involves annotation of the data set, text pre-processing, and word vectorization using the Word2Vec skip-gram model. Sentence vectors are then constructed by averaging word vectors.

For classification, we employ ensemble machine learning, specifically Random Forest, which is widely used in sentiment analysis research, such as in [10–14]. Our study is conducted on five labeled datasets derived from YouTube video comments with varying class imbalance ratios.

Our study aims to prove the increase in the precision of sentiment analysis if combining Word2vec and FastText vectorization with SMOTE, Borderline SMOTE, and ADASYN to identify the best combination of word embedding and oversampling.

The novelty of this research lies in identifying the optimal combination of word embedding and oversampling techniques to improve the accuracy of sentiment analysis. In addition, this study quantifies the improvement in accuracy achieved through these combined methods.

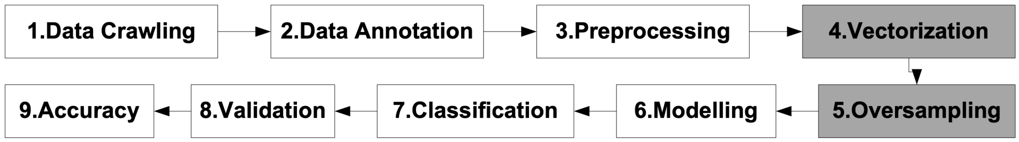

Figure 1: Research steps.

2 Research Method

This section discusses research steps, data crawling and annotation, pre-processing, vectorization, SMOTE process, oversampling, building the model and classification process, and validation and accuracy.

2.1 Research Steps

The research began with the crawling of social media data. The collected dataset was then manually annotated by experts before being pre-processed. The next step was vectorization, a crucial phase in this study, in which Word2Vec and FastText were tested. Another key aspect of this research was the evaluation of the most effective oversampling method (SMOTE, Borderline SMOTE, or ADASYN). The final phase involved validating the accuracy of the model through machine learning-based modeling, classification testing, validation, and accuracy evaluation Figure 1.

2.2 Data Crawling and Annotation

The opinion dataset was collected from comments on YouTube videos that discuss the debates between Indonesian presidential candidates for 2019. The dataset was labeled by an expert [15] using a voting mechanism to classify sentiments into positive and negative polarities. As a result, expert-labeled sentiment annotations exhibited an imbalanced distribution. Table 1 summarizes the dataset obtained from crawling and labeling, organized based on the comparison of the ratio between the majority and minority classes.

The annotation results show an imbalance in the number of positive and negative comments. Table 1 presents a summary of the crawled and labeled data sorted by comparison of the ratio of the majority class and the minority class.

Table 1: Datasets information

Dataset Number	**Number of Instances**	**Sentiment Polarity**		**Class Ratio**
		Positive	*Negative*	
1	5134	3572	1562	2.29:1
2	13061	9262	3799	2.44:1
3	29520	21253	8267	2.6:1
4	11323	8417	2906	2.9:1
5	25694	22205	3489	6.36:1

2.3 Pre-Processing

The preprocessing stage begins with data cleaning, which includes removing URLs, numbers, and single characters, converting text to lowercase, replacing emoticons with their textual equivalents, removing non-alphabetic characters, eliminating stop words, and tokenizing. We use a dictionary of slang words containing 5,123 words, where the detected slang words are converted to their standard forms. The final step is stemming, which removes affixes from each word.

2.4 Vectorization

The limitations of the Bag-of-Words (BoW) and TF-IDF methods stem from their generation of high-dimensional matrix vectors. BoW converts a set of sentences into a vector representing term frequencies, where each word within a sentiment category is counted [16]. The word that occurs most frequently in a particular sentiment type is then assigned to the corresponding sentiment group. The size of the matrix is determined by the product of the number of words and the instances of the dataset. Unlike BoW and TF-IDF, word embedding methods such as Word2Vec generate lower-dimensional matrices, where the dimensionality depends on the predefined number of features and words. Sentences in the dataset are transformed into vectors by averaging the vector representations of their constituent words. Consequently, the size of the opinion matrix is calculated as the number of features multiplied by the number of opinions. The resulting sentence vectors are subsequently used in the oversampling process. FastText is an advance of Word2Vec that performs vectorization in a more granular manner. Unlike Word2Vec, which treats each word as an atomic unit, FastText represents words as a combination of character-level n-grams, where n can vary from one to the total length of the word [17]. This approach makes FastText more detailed, albeit computationally more intensive and requiring a larger dictionary. One key advantage of FastText is its ability to generate vector representations for out-of-vocabulary words by approximating them based on similar character n-grams.

2.5 SMOTE Process

SMOTE results in a number of new distances along the correlation line between feature x and feature y. As shown in Figure 2, initially, there were 15 green data (majority) and seven red data (minority). SMOTE will make a virtual relationship line between adjacent feature instances in a minority class and add the synthetic data between the instance features in minority data. This is done until the number of instance data is balanced with the majority of data.

2.6 Oversampling

The sentence vectors of the minority class were added to balance using SMOTE, Borderline SMOTE, and ADASYN. The test was carried out in rotation, from dataset 1, 2, 3, and 4 to dataset 5. The results of the oversampling become training data. Training data were used here for modeling and test data for model testing.

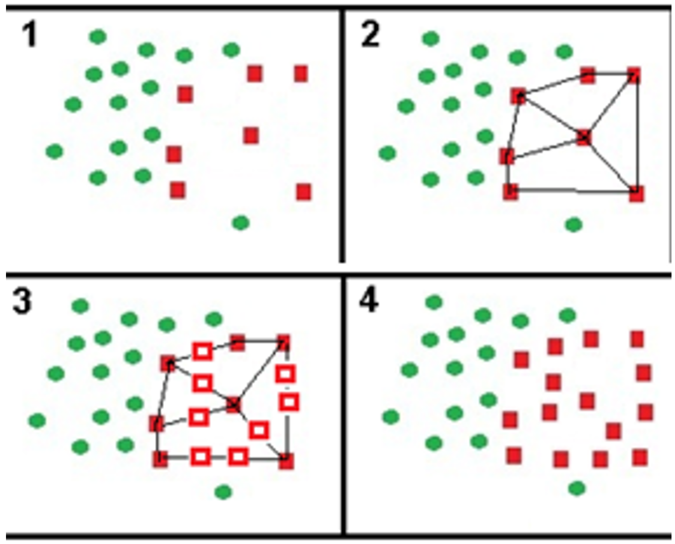

Figure 2: SMOTE process.

2.7 Building the Model and Classification Process

Random Forest (RF) was chosen as the modeling technique due to its track record of delivering high accuracy and reduce overfitting, as supported by previous research [18]. RF mitigates overfitting by generating several trees and employing bootstrapping. Each tree contributes to the classification results, and the ultimate classification is determined by the majority class among these trees. Training and testing data were split into an 80:20 ratio.

2.8 Validation and Accuracy

The performance of all sentiment analysis models in this research was measured using the confusion matrix. Research on sentiment analysis commonly uses the confusion matrix, as in [19], [20]. The confusion matrix compares the results of the actual class classification and the prediction class of the results of the classifier model. The number of confusion matrix would be the variable to calculate the level of accuracy, precision, recall, and the F-measure values. Table 2 shows the confusion matrix for the prediction of two classes.

Table 2: Confusion matrix

		Actual Class	
		Positive	Negative
Predicted Class	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

The accuracy of the confusion matrix was measured, that is, how many instances were correctly classified by the sentiment analysis model. The correct classification was true positive (TP) and false negative (FN). The accuracy measurement used (1).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Precision was used to measure the accuracy of the classifier to measure the true positive, in which the high precision meant fewer false positives (FP), while the low precision meant more false positives (FP). The formula of precision used (2).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

The recall measured the completeness or the classifier sensitivity. The formula of recall was in (3), in which the high value of recall meant fewer false negatives (FN), while the lower recall meant more false negatives (FN). The increase of the recall values often reduces the precision values as the classifier becomes more difficult to be accurate with the increase of sample spaces.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

The precision and recall were used to calculate the F-measure as the average of weighted harmonic from the precision and recall. The formula of F-measure was in (4) and the measurement of F-measure was also useful as the accuracy.

$$\text{F}-\text{ measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

To measure the performance of the oversampling methods, we measured the accuracy and F-measure in the baseline condition as an initial condition of each dataset that was only given with the word embedding process (Word2Vec and FastText). We measured the accuracy and F-measure in the baseline condition, and then we measured the accuracy and F-measure after conducting SMOTE, Borderline SMOTE, and ADASYN. We identify the increase in accuracy after conducting the oversampling as the parameter of performance of the oversampling methods.

3 Results

Parameters of Word2vec and FastText used were determined by word vector dimensionality (or number of features) = 50, minimum word count = 5, number of workers/parallel threads = 4, and context window size = 5.

Meanwhile, the results of the oversampling made the size of each dataset changed because the number of instances of minority class was increased to be equal with the majority class as shown in Table 3.

Table 3: The change of the number of instances before and after oversampling

Dataset	Number of Comment					
	Before Oversampling			After Oversampling		
	Positive	Negative	Total	Positive	Negative	Total
D1	3,572	1,562	5,134	3,572	3,572	7,144
D2	9,262	3,799	13,061	9,262	9,262	18,524
D3	21,253	8,267	29,520	21,253	21,253	42,506
D4	8,417	2,906	11,323	8,417	8,417	16,834
D5	22,205	3,489	25,694	22,205	22,205	44,410

3.1 Comparing the Performance of Word Embedding to the Imbalanced Data (Baseline)

This experiment aimed to evaluate the accuracy and F-measure in the initial testing phase using only word embedding. Additionally, it assessed the performance of word embedding under imbalanced conditions (baseline). These results served as baseline accuracy and F-measure values for comparison with the results obtained after applying oversampling techniques. The calculated results are presented in Table 4.

Table 4: Accuracy and F-measure in the baseline condition (Imbalance)

Dataset	Imbalance Condition (Baseline)			
	Accuracy		F-measure	
	Word2Vec	FastText	Word2Vec	FastText
Dataset 1	68.5%	66.2%	65.7%	62.8%
Dataset 2	73.6%	73.6%	70.8%	71.0%
Dataset 3	74.2%	76.0%	70.3%	72.8%
Dataset 4	74.2%	77.9%	69.8%	73.9%
Dataset 5	87.0%	86.5%	82.0%	81.5%

As shown in Table 4, dataset 1 and 5, the Word2vec method was more excellent. In contrast, in datasets 3 and 4, FastText was found to be more excellent than Word2vec. Thus, the performance of FastText was more excellent compared to Word2vec to support the results of the more accurate classification.

3.2 Comparing the Accuracy of Oversampling Method Classification After Word embedding

Table 5 until Table 8 shows the results of our experiments. We have conducted 60 experiments as the test of the combination of 2 (two) types of word embedding (Word2Vec and FastText), 3 (three) oversampling conditions (SMOTE, Borderline-SMOTE, and ADASYN), measured at 2 (two) validation values: accuracy and F-measure on five datasets. The baseline condition referred to the accuracy/F-measure of the classification on the imbalance dataset without implementing the oversampling method. In Table 5 to Table 8, we highlighted the numbers of the increase in accuracy more than 10% from the baseline.

3.2.1 Analysis on the accuracy of oversampling results after Word2Vec

The accuracy and F-measure values for the model combining Word2Vec with SMOTE, Borderline-SMOTE, and ADASYN were evaluated. Table 5 summarizes the accuracy results and the corresponding improvements for each model and dataset compared to the baseline.

As shown in Table 5, the Borderline-SMOTE method achieved the highest accuracy among the three oversampling techniques. Notably, dataset 4 exhibited an accuracy increase of more than 10% (10.1%). However, the highest overall accuracy was observed in dataset 5 when using ADASYN and SMOTE. Table 6 presents the F-measure results for the combination of Word2Vec with the three oversampling methods.

Table 5: Accuracy of the model using the combination of Word2Vec and oversampling

Dataset	Baseline	Accuracy and Improvement From Baseline					
		Borderline-SMOTE		ADASYN		SMOTE	
	Acc %	Acc %	↑ %	Acc %	↑ %	Acc %	↑ %
(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
D1	68.5	75.4	6.9	75.5	7.0	75.6	7.1
D2	73.6	83.1	9.5	81.3	7.7	81.5	7.9
D3	74.2	83.4	9.2	83.0	8.8	83.5	9.3
D4	74.2	84.3	**10.1**	82.9	8.7	83.1	8.9
D5	87.0	**91.0**	4.0	**91.3**	4.3	**91.3**	4.3

Table 6: F-measure model using the combination of Word2Vec and oversampling

Dataset	Baseline	F-measure and Improvement From Baseline					
		Borderline-SMOTE		ADASYN		SMOTE	
	F1-S%	F1-S%	↑ %	F1-S%	↑ %	F1-S%	↑ %
(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
D1	65.7	75.3	9.6	75.5	9.8	75.5	9.8
D2	70.8	83.0	**12.2**	81.1	**10.3**	81.5	**10.7**
D3	70.8	83.3	**13.0**	82.9	**12.6**	83.5	**13.2**
D4	69.8	84.2	**14.4**	82.8	**13.0**	83.1	**13.3**
D5	82.0	**91.0**	9.0	**91.3**	9.3	**91.3**	9.3

As shown in Table 6, it can be seen that the Borderline-SMOTE method more dominantly provided highest F-measure and accuracy of the ADASYN and SMOTE methods. Overall, the increase in F-measure value was greater than 10 % found in dataset 2, dataset 3, and dataset 4.

3.2.2 The analysis of accuracy of the oversampling results after FastText

Table 7 presents the increase in accucary of SMOTE, Borderline-SMOTE and ADASYN after the FastText method and their improvement from baseline. As seen in Table 7, Borderline-SMOTE provided a higher accuracy than ADASYN and SMOTE, although it did not reach more than 10 %. However, in general, the accuracy of all FastText models was almost equal to Word2Vec but, in this case, the increase in the accuracy of the FastText from the baseline was lower. Table 8 shows the results of F-Measure on FastText models.

Table 7: Model accuracy using a combination of FastText and oversampling

Dataset	Baseline	F-measure and Improvement From Baseline					
		Borderline-SMOTE		ADASYN		SMOTE	
	Acc %	Acc %	↑ %	Acc %	↑ %	Acc %	↑ %
(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
D1	66.2	76.1	9.9	75.5	9.3	76.1	9.9
D2	73.6	82.1	8.5	82.3	8.7	81.2	7.6
D3	76.0	84.0	8.0	82.5	6.5	83.3	7.3
D4	77.9	84.1	6.2	83.8	5.9	83.2	5.3
D5	86.5	**91.3**	4.8	**90.6**	4.1	**91.0**	4.5

Table 8: F-measure model using the combination of FastText and oversampling

Dataset	Baseline	F-measure and Improvement From Baseline					
		Borderline-SMOTE		ADASYN		SMOTE	
	F1-S%	F1-S%	↑ %	F1-S%	↑ %	F1-S%	↑ %
(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
D1	62.8	76.1	**13.3**	75.4	**12.6**	76.1	**13.3**
D2	71.0	82.0	**11.0**	82.2	**11.2**	81.2	**10.2**
D3	72.8	84.1	**11.3**	82.5	9.7	83.2	**10.4**
D4	73.9	84.0	**10.1**	83.7	9.8	83.1	9.2
D5	81.5	**91.3**	9.8	**91.0**	9.5	**91.0**	9.5

As seen in Table 8, from all experiments, the Borderline-SMOTE method provided the highest F-measure compared to ADASYN and SMOTE methods. The increase in the F-measure value was more than 10 percent in datasets 1, 2, 3, and 4. Then, it was followed by SMOTE and ADASYN. The increase in F-measure on dataset 5 was found at the lowest; this was related to the high ratio of majority and minority, making the synthetic data variation not as good as the one in the majority class. Based on all the research results, the selection of the oversampling and word embedding methods affected the accuracy and F-measure. The condition of the dataset also affected its accuracy, particularly on the initial majority and minority ratios of the dataset.

4 Discussion

The experimental results provide valuable insights into the effectiveness of different word embedding techniques and oversampling methods in handling imbalanced sentiment classification tasks. The discussion is structured around three key findings: the impact of word embedding methods on classification performance, the role of oversampling in improving classification metrics, and the comparison between different oversampling techniques.

4.1 Impact of Word Embedding on Classification Performance

The results of the baseline condition (Table 4) indicate that Word2Vec and FastText exhibit varying performance in different datasets. Word2Vec demonstrated higher accuracy and F-measure values in datasets 1 and 5, while FastText outperformed Word2Vec in datasets 3 and 4. This suggests that the performance of word embedding techniques is dataset-dependent, possibly influenced by the vocabulary distribution, text length, and contextual dependencies within each dataset. The higher performance of FastText in certain datasets can be attributed to its ability to capture subword information, which may be beneficial in datasets with complex linguistic structures.

4.2 The Role of Oversampling in Improving Classification Metrics

The application of oversampling techniques (SMOTE, Borderline-SMOTE, and ADASYN) led to significant improvements in classification performance across all datasets. As observed in Tables 5-8, oversampling improved both accuracy and F-measure compared to the baseline condition. The most notable improvements were seen in datasets where the class imbalance was more pronounced, such as datasets 2, 3, and 4. This demonstrates that addressing class imbalance is crucial in sentiment classification tasks, as it allows the classifier to learn more representative features of both classes, leading to more balanced decision boundaries.

4.3 Comparison of Oversampling Methods

Among the three oversampling methods, Borderline-SMOTE consistently provided the highest accuracy and F-measure values in both the Word2Vec and FastText models. This suggests that generating synthetic samples near the decision boundary of the minority class effectively enhances the classifier's ability to distinguish between classes. In particular, dataset 4 showed the highest accuracy improvement of 10.1% using Borderline-SMOTE with Word2Vec. Similarly, the F-measure improvements exceeded 10% in datasets 2, 3, and 4, reinforcing the effectiveness of Borderline-SMOTE in handling imbalanced sentiment data. ADASYN and SMOTE also contributed to performance improvements, with ADASYN showing slightly higher gains than SMOTE in some cases. However, the increase in accuracy and F-measure was generally lower compared to Borderline SMOTE. This implies that while these techniques help mitigate class imbalance, they may not be as effective in refining the decision boundaries as Borderline-SMOTE.

4.4 Comparison Between Word2Vec and FastText with Oversampling

When comparing the impact of oversampling on the Word2Vec and FastText models, it was found that Word2Vec generally achieved higher performance improvements with oversampling compared to FastText. Although FastText showed competitive baseline performance, the improvements in accuracy and F-measure after applying oversampling were slightly lower. This suggests that while FastText effectively captures subword information, its representation might not be as sensitive to oversampling-induced changes as Word2Vec.

4.5 Practical Implications and Future Work

The findings of this study highlight the importance of selecting appropriate word embedding and oversampling techniques in sentiment classification tasks. Borderline SMOTE emerged as the most effective oversampling method, particularly in scenarios with severe class imbalance. Future research could explore hybrid approaches that combine multiple oversampling techniques or investigate deep learning-based generative methods for synthetic sample creation. Additionally, evaluating the impact of different word vector dimensions and contextual embeddings (e.g., BERT) could provide deeper insights into optimizing sentiment classification performance.

5 Conclusion

The study aims to prove an increase in the accuracy of sentiment analysis when combining Word2Vec and FastText with SMOTE, Borderline-SMOTE, and ADASYN. In the baseline condition, FastText found the word embedding with higher accuracy. Meanwhile, the best oversampling method was Borderline-SMOTE, which, in almost all experiments, provided the highest increase in accuracy compared to other methods. The best combination of word embedding and oversampling for accuracy was found in Word2Vec and Borderline-SMOTE. The combination of FastText and Borderline-SMOTE became the proper model to obtain the maximum F-measure value. The next research is to add to the test dataset from other opinion sources (Twitter, Instagram, Facebook). In addition, there are other word embedding and oversampling methods that can be examined for their performance. Several types of machine learning can be used to identify their effect on model performance.

Acknowledgments

This study is financially supported by Universitas Pembangunan Nasional "Veteran" Yogyakarta by the scheme of "Penelitian Internal Dasar". The authors would like to thank the Lembaga Penelitian dan Pengabdian Masyarakat and Department of Informatics, Universitas Pembangunan Nasional "Veteran" Yogyakarta Indonesia for their support for this research.

References

[1] M. Kumar, L. Khan, and H.-T. Chang, "Evolving techniques in sentiment analysis: a comprehensive review," *PeerJ Computer Science*, vol. 11, p. e2592, 2025.

[2] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, pp. 1–30, 2018.

[3] M. R. Pribadi, D. Manongga, H. D. Purnomo, I. Setyawan *et al.*, "Sentiment analysis of the pedulilindungi on google play using the random forest algorithm with smote," in *2022 International Seminar on Intelligent Technology and Its Applications (ISITIA)*. IEEE, 2022, pp. 115–119.

[4] I. N. Switrayana, D. Ashadi, H. Hairani, and A. Aminuddin, "Sentiment analysis and topic modeling of kitabisa applications using support vector machine (svm) and smote-tomek links methods," *International Journal of Engineering and Computer Science Applications (IJECSA)*, vol. 2, no. 2, pp. 87–98, 2023.

[5] C. S. K. Aditya, G. W. Wicaksono, and H. A. S. Heryawan, "Sentiment analysis of the 2024 presidential candidates using smote and long short term memory," *Jurnal Informatika Universitas Pamulang*, vol. 8, no. 2, pp. 279–286, 2023.

[6] L. Cahyaningrum, A. Luthfiarta, and M. Rahayu, "Sentiment analysis on the impact of mbkm on student organizations using supervised learning with smote to handle data imbalance," *Inform: Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi*, vol. 9, no. 1, pp. 58–66, 2024.

[7] A. Glazkova, "A comparison of synthetic oversampling methods for multi-class text classification," *arXiv preprint arXiv:2008.04636*, 2020.

[8] E. Biswas, K. Vijay-Shanker, and L. Pollock, "Exploring word embedding techniques to improve sentiment analysis of software engineering texts," in *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 2019, pp. 68–78.

[9] N. Cahyana, S. Khomsah, and A. S. Aribowo, "Improving imbalanced dataset classification using oversampling and gradient boosting," in *2019 5th International Conference on Science in Information Technology (ICSITech)*. IEEE, 2019, pp. 217–222.

[10] Y. Al Amrani, M. Lazaar, and K. E. El Kadiri, "Random forest and support vector machine based hybrid approach to sentiment analysis," *Procedia Computer Science*, vol. 127, pp. 511–520, 2018.

[11] M. A. Fauzi, "Random forest approach fo sentiment analysis in indonesian," *Indones. J. Electr. Eng. Comput. Sci*, vol. 12, pp. 46–50, 2018.

[12] G. Khanvilkar and D. Vora, "Sentiment analysis for product recommendation using random forest," *International Journal of Engineering & Technology*, vol. 7, no. 3, pp. 87–89, 2018.

[13] Y. Hegde and S. Padma, "Sentiment analysis using random forest ensemble for mobile product reviews in kannada," in *2017 IEEE 7th international advance computing conference (IACC)*. IEEE, 2017, pp. 777–782.

[14] N. Saleena *et al.*, "An ensemble classification system for twitter sentiment analysis," *Procedia computer science*, vol. 132, pp. 937–946, 2018.

[15] L. Shyamasundar and P. Jhansi Rani, "A multiple-layer machine learning architecture for improved accuracy in sentiment analysis," *The Computer Journal*, vol. 63, no. 3, pp. 395–409, 2020.

[16] S. Kaur, P. Kumar, and P. Kumaraguru, "Automating fake news detection system using multi-level voting model," *Soft Computing*, vol. 24, no. 12, pp. 9049–9069, 2020.

[17] B. Kuyumcu, C. Aksakalli, and S. Delil, "An automated new approach in fast text classification (fasttext) a case study for turkish text classification without pre-processing," in *Proceedings of the 2019 3rd international conference on natural language processing and information retrieval*, 2019, pp. 1–4.

[18] A. Aribowo, H. Basiron, N. Herman, and S. Khomsah, "Fanaticism category generation using tree-based machine learning method," in *Journal of Physics: Conference Series*, vol. 1501, no. 1. IOP Publishing, 2020, p. 012021.

[19] A. Jain and V. Jain, "Sentiment classification of twitter data belonging to renewable energy using machine learning," *Journal of information and optimization sciences*, vol. 40, no. 2, pp. 521–533, 2019.

[20] M. Bilgin and İ. F. Şentürk, "Sentiment analysis on twitter data with semi-supervised doc2vec," in *2017 international conference on computer science and engineering (UBMK)*. Ieee, 2017, pp. 661–666.