JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

EFFECTIVENESS OF WORD2VEC AND TF-IDF IN SENTIMENT CLASSIFICATION ON ONLINE INVESTMENT PLATFORMS USING SUPPORT VECTOR MACHINE

Fadil Rifaldy*[1)], Yuliant Sibaroni[2)], Sri Suryani Prasetiyowati[3)]
1. Telkom University, Indonesia
2. Telkom University, Indonesia
3. Telkom University, Indonesia

Article Info
Keywords: Sentiment Classification; Investment App; Word2Vec; TF-IDF; Support Vector Machine; Nanovest

Article history:
Received 24 Oktober 2024
Revised 17 November 2024
Accepted 7 Desember 2024
Available online 15 March 2025

DOI :
https://doi.org/10.29100/jipi.v10i2.6055

Corresponding Author.
Fadil Rifaldy
E-mail address:
fadilrifaldy@student.telkomuniversity.ac.id

ABSTRACT
Investing in Indonesia is increasingly popular, especially among the millennial generation. investments such as deposits, gold, stocks, and online investment applications are increasingly in demand. This research focuses on the sentiment classification of user reviews of the Nanovest online investment application on the Google Play Store using the Support Vector Machine (SVM) method. SVM is used because it can classify opinions into positive and negative sentiment classes with good accuracy, by evaluating how effective Word2Vec features extraction that can convert words in a text into numerical vectors and TF-IDF that is capable of high-dimensional word weighting and TF-IDF Weighted Word2Vec combination features to produce richer vector representations. Tests were conducted using four SVM kernels namely Linear, Polynomial, RBF, and Sigmoid. The results show that Word2Vec with RBF kernel and 300 vector size produces the highest accuracy of 95.46%, the combination of TF-IDF Weighted Word2Vec also gives good performance with 95.29% accuracy on RBF kernel. However, TF-IDF alone resulted in the lowest accuracy of 93.31% on the Sigmoid kernel. This research shows that Word2Vec and combined feature extraction methods are effective in improving sentiment classification performance compared to TF-IDF.

.

I. INTRODUCTION

IN today's technological era, economic factors, where activities such as buying, selling, transactions, and investments are increasingly being carried out online and are promising opportunities [1], [2]. Investment is the allocation of funds to buy financial instruments or assets to generate profits in the form of dividends or interest [3].

Popular investments in Indonesia according to HSBC include time deposits, gold, property, stocks, mutual funds, and peer-to-peer lending [4]. The 2017 HSBC Media Advisory survey shows that 39% of millennials in Indonesia are very interested in taking risks in investment [3]. In the fourth quarter of 2020, realized investment reached IDR 214.7 trillion, an increase of 3.1% compared to the previous year, reflecting the increasing public interest in investing [4]. Also supported by the emergence of various online investment applications, but the advantages and disadvantages become the benchmark for the quality of application services. By analyzing sentiment, we can find out customer views and identify potential problems or areas of improvement [1], [5].

Research related to sentiment analysis using SVM has been conducted by several researchers [10], [7], [11], [12]. The Support Vector Machine (SVM) is a widely used algorithm that can accurately classify opinions into positive, and negative sentiment categories) [6], [7], [8]. SVM classifies by creating a hyperplane that distinguishes between positive and negative opinions [6]. To make it easier to deal with dimensionality issues, SVM changes information to higher aspects utilizing kernel operations like the Linear, Radial Basis Function (RBF), Polynomial, and Sigmoid [9].

Research [10] analyzed the sentiment of Ruangguru application reviews using SVM classification, and testing using a Linear kernel resulted in 89.7% accuracy. The combination of training data and test data resulted in 90% accuracy. With K-Fold, the highest accuracy is at K-Fold 6, 9, and 10 which is 90.2%. The accuracy rate in this study is high in the range of 90%. Meanwhile, research [7] analyzes the sentiment of online transportation application reviews using SVM classification with RBF kernel and Word2Vec feature extraction with Skip-gram model, size 100 dimensions, and window 5. The test results show quite good performance, namely for the Gojek application the accuracy is 87%. While in the Grab application, the accuracy is 82%. In research [11], sentiment

classification on application reviews of providers by.U uses SVM and TF-IDF as feature extraction. TF-IDF + SVM with 5-fold Validation produces a good accuracy of 84.7%, precision of 84.9%, recall of 84.7%, and f-measure of 84.8%, the fold 2 results have the highest accuracy, at 86.1%. Research [12] analyzed sentiment on film reviews using the Term Frequency-Inverse Document Frequency (TF-IDF) method and a classification algorithm, namely Support Vector Machine (SVM). The choice of this method is due to its ability to give weight to words and classify data with high dimensions. The use of more training data will have an impact on system performance. Based on the test scenarios conducted, the results show that the combination of TF-IDF and SVM algorithms is effective for movie review cases, with 85% accuracy, 100% precision, 70% recall, and F1-Score value reaching 82%.

Based on these research results, feature extraction plays an important role in improving the accuracy of sentiment classification models. This process aims to capture important information from the text and convert it into a numerical representation that is useful for classification algorithms. Effective feature extraction helps the model to better understand and interpret the text data, thus improving its performance in sentiment classification. In these studies, TF-IDF and Word2Vec have demonstrated their ability to improve the accuracy of sentiment classification models. TF-IDF calculates the frequency of each word in a document (TF) and compares it to the frequency of that word in the entire collection of documents (IDF). Words that appear frequently in certain documents but rarely appear in other documents will get a higher weight, thus helping the model to focus on sentiment-relevant words. On the other hand, Word2Vec converts words into numerical vectors representing their semantic meaning by building a vocabulary from the training text data and learning the vector representation of each word in the corpus. The resulting vectors can be used as features to help the model understand the contextual relationships of sentences. Word2Vec does this by mapping words that share the same context into a contiguous vector space, thus allowing the model to recognize more complex patterns and relationships in the text data. This research will evaluate the effectiveness of Word2Vec Word Embedding and TF-IDF as well as the features of the TF-IDF weighted Word2Vec combination as these two techniques complement each other by making them work better together than working individually. The combination of TF-IDF and Word2Vec is expected to create a richer vector representation, where TF-IDF weights are used to adjust Word2Vec vectors by multiplying each Word2Vec vector with TF-IDF values. This approach allows the model to consider the frequency and relevance of words in the document through TF-IDF weights, as well as their semantic meaning through Word2Vec vectors. Thus, this combination is expected to improve the accuracy and effectiveness of the model in sentiment classification.

The classification algorithm used is SVM due to its high accuracy in classifying opinions into high-dimensional positive and negative sentiment classes, outperforming other commonly used algorithms such as Naive Bayes and Decision Tree. SVM excels at finding the optimal hyperplane that maximizes the margin between different classes, thus improving generalization to unseen data. SVM effectively handles the complexity of the feature space with a kernel function, this function can manage non-linear relationships by transforming data into a higher dimensional space and capturing complex sentiment patterns. This contrasts with Naive Bayes which assumes feature independence, and Decision Tree which is prone to overfitting. SVM parameter optimization is essential to improve model performance. Parameters such as kernel, C, and gamma need to be adjusted to ensure the model handles the data effectively and produces accurate classifications. This optimization process helps find the best combination that maximizes the margin between different classes, thereby reducing the risk of overfitting and underfitting. This study demonstrates a rigorous and careful approach to ensure that classification models can effectively handle complex data sets and improve accuracy. Performance evaluation is needed to find the most effective and optimal model and the expected results can contribute to the field of sentiment analysis, especially in knowing the level of user satisfaction of the Nanovest application based on the sentiment classification of user reviews of the application.

II. Research Methodology

This study's system flow, which includes datasets, preprocessing, TF-IDF feature extraction, Word2Vec feature extraction, Word2Vec weighted TF-IDF combination feature extraction, learning of SVM model, and evaluation, is shown in Figure 1.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

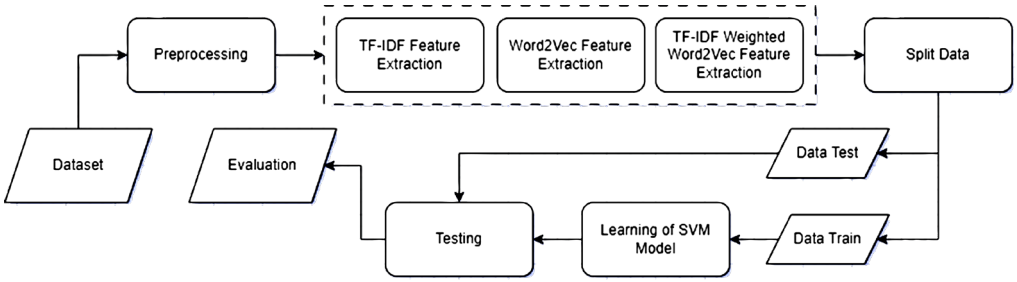

Fig. 1. System Flow

A. Dataset

The dataset used in this research is a review of the Nanovest application on the Google Play Store which collected around 10.000 data by scrapping data using Python. Data labeling will be done using Manual Annotation or manually by reading the dataset reviews, and then determining whether the reviews have positive or negative sentiments. The results obtained are more precise and accurate and do not require computation [13]. An example dataset can be seen in Table I.

TABLE I
RESEARCH DATASETS

Class	Label	Sentences
Positive	1	Aplikasi nanovest emang aman bgt buat inverstasi crypto dan saham pokonya sangat terpercaya dan dapat diandalkan KERENN!!!
Negative	0	Aplikasi ga jelas, setiap masuk platform selalu keluar sendiri... mungkin inilah yg dinamakan buat aplikasi tapi masih rentan segala hal... Jgn down-load...!!!

B. Preprocessing

Preprocessing is the first stage performed before classification. This stage is carried out to process raw datasets that are unstructured and not ready to pass the classification stage. The flowchart of the Preprocessing stage used can be seen in Figure 2.

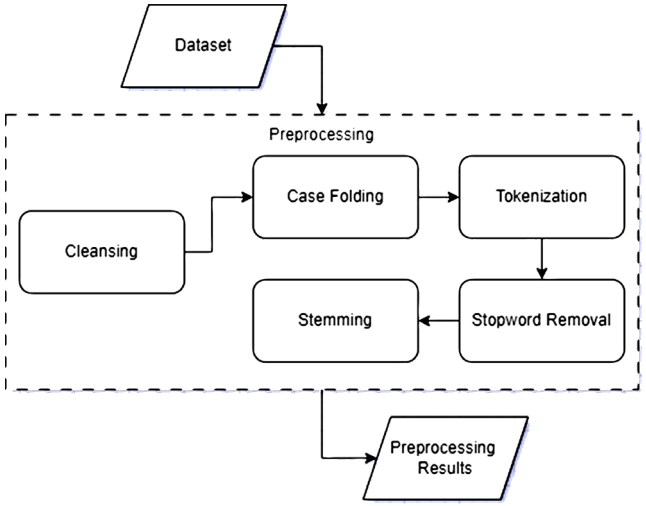

Fig. 2. Preprocessing Flow

1) Cleansing

This stage is the stage of removing non-alphabetic characters to reduce noise. The alphabets removed are punctuation marks such as periods (.), commas (,), question marks (?), and exclamation marks (!), as well as symbols such as the '@' sign for usernames, hashtags (#), emoticons, and website addresses.

2) Case Folding

Case Folding is a stage to convert alphabetic characters containing capital letters (uppercase) that have gone through the cleaning stage into lowercase letters (lowercase). This process helps in overcoming problems when words or phrases are written with different capital letters.

3) Tokenization

Tokenization is a stage where the sentence is broken into separate words, tokenization is used to separate a sentence into parts or what are commonly called tokens so that sentences can be separated and can be distinguished.

4) Stopword Removal

Stopword Removal is the process of removing common words that often appear in the text but have no significance for text analysis or modeling such as "yang", "untuk", "pada", "ke", "dan", "ini", "because", and others in the stopword dictionary. The stopword dictionary used comes from the Sastrawi library, an open-source library that provides various functions for natural language processing (NLP) in Indonesian. The list of words in the stopword dictionary used can be seen in Table V.

TABLE V
STOPWORD DICTIONARY

No.	Stopword List	No.	Stopword List	No.	Stopword List	No.	Stopword List	No.	Stopword List	No.	Stopword List
1.	ada	20.	adalah	39.	agar	58.	agak	77.	akan	94.	amat
2.	anda	21.	anu	40.	antara	59.	apakah	78.	apalagi	95.	atau
3.	bahwa	22.	bagi	41.	bagaimanapun	60.	banyak	79.	begitu	96.	belum
4.	berbagai	23.	bila	42.	bisa	61.	boleh	80.	dan	97.	dapat
5.	dari	24.	daripada	43.	dahulu	62.	dalam	81.	demi	98.	demikian
6.	dia	25.	dll	44.	dengan	63.	di	82.	dst	99.	dua
7.	dulunya	26.	dsb	45.	hanya	64.	hal	83.	ini	100.	itu
8.	ia	27.	iya	46.	ingin	65.	jadi	84.	jika	101.	juga
9.	jadi	28.	jikalau	47.	kali	66.	kalian	85.	kalau	102.	kami
10.	karena	29.	kemudian	48.	kepada	67.	kita	86.	klien	103.	ke
11.	lain	30.	lagi	49.	melainkan	68.	mari	87.	maka	104.	mengapa
12.	mereka	31.	menjadi	50.	melalui	69.	menurut	88.	masa	105.	masih
13.	memang	32.	nanti	51.	nah	70.	namun	89.	padahal	106.	para
14.	pada	33.	pun	52.	sampai	71.	saja	88.	sambil	107.	sana
15.	sangat	34.	selama	53.	sebab	72.	serta	89.	sebagai	108.	sedikit
16.	selain	35.	seseorang	54.	sama	73.	seperti	90.	setelah	109.	sekitar
17.	sebelum	36.	setiap	55.	sehingga	74.	sesuatu	91.	sudah	110.	supaya
18.	sedangkan	37.	seraya	56.	tentu	75.	tanpa	92.	tentang	111.	toh
19.	untuk	38.	waktu	57.	walau	76.	yakni	93.	yang	112.	yaitu

5) Stemming

Stemming removes prefixes, inserts, suffixes, and con-fixes (combinations of prefixes and suffixes) from derived words to locate the word's root. With stemming, words with a similar root will be viewed as a similar token (include). This contributes to improved data retrieval performance in information retrieval. The Sastrawi library, an open-source library that offers a variety of natural language processing (NLP) functions in Indonesian, includes stemming as one of its primary functions. Examples of stemming can be seen in Table VI.

TABLE VI
STEMMING

Input	Output
Aplikasinya memudahkan pemula yang mau membeli bitcoin, tampilannya juga sederhana jadi membantu banget	['aplikasi', 'mudah', 'mula', 'yang', 'mau', 'beli', 'bitcoin', 'tampil', 'juga', 'sederhana', 'jadi', 'bantu', 'banget']

C. Feature Extraction Term Frequency – Inverse Document Frequency (TF-IDF)

Term Frequency-Inverse Document Frequency (TF-IDF) is a method to determine word weights based on document frequency (DF) [14], [15]. The terms Term Frequency and Inverse Document Frequency are combined to form TF-IDF [16]. This method combines the Term Frequency (TF) value, which reflects the number of occurrences of a word in a document, with the Inverse Document Frequency (IDF) value, which calculates the logarithmic inverse of the document frequency where the word is searched [17].

The Term Frequency-Inverse Document Frequency (TF-IDF) calculation can be seen in equations (1), (2), and (3) [18].

1) Term Frequency (TF)

$$TF_t = (t, d) \tag{1}$$

The Term Frequency (TF) value is obtained from the frequency of occurrence of feature *t* in document *d* [18].

2) Inverse Document Frequency (IDF)

$$IDF_t = \log \frac{n}{df(t)} + 1 \tag{2}$$

The Inverse Document Frequency (IDF) value is obtained from the logarithm of the number of documents *n* divided by the *df* of documents containing feature *t* [18].

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

3) Term Frequency - Inverse Document Frequency (TF-IDF)

$$W_t = TF_t \times IDF_t \quad (3)$$

The *Term Frequency - Inverse Document Frequency* (TF-IDF) (W_t) is obtained by multiplying the *TF* value with *IDF* [18].

D. Feature Extraction Word2Vec

Word2Vec is a machine-learning technique for converting text words into numerical vectors, which can be used for text analysis or other natural language processing (NLP) purposes [19]. Word2Vec was proposed by Mikolov in 2013, designed to convert an input corpus into an output vector [20].

In the initial stage of the Word2Vec process, the first step involves forming a vocabulary from the training text data. After that, the process learns a vector representation of each word in the corpus. The resulting vectors may be useful features in machine learning and natural language processing applications [21]. Continuous Bag of Words (CBOW) and Skip-gram are the two primary algorithms in Word2Vec [19].

1) Continuous Bag of Words (CBOW)

CBOW is a neural network-based architecture model in which the non-linear hidden layer is removed, and the projection layer is distributed across words. CBOW projects a word by considering other words in the same sentence. Unlike bag-of-words models, CBOW uses a continuously distributed representation [20]. CBOW predicts something about the vocabulary w(t) from the context w(t -2), w(t -1), w(t +1), w(t +2) as in the Continuous Bags-of-Words (CBOW) architecture model which can be seen in Figure 3 [20].

2) Skip-gram

The goal of the Skip-gram model is to provide a word representation that is effective in predicting nearby words in a sentence or document. Skip-gram performs prediction by considering the word before and after the word is processed [21]. The Skip-gram model produces outputs w(t-2), w(t-1), w(t+1), and w(t +2) because it is specifically designed to estimate the surrounding context for a given word w(t) to maximize its prediction accuracy. Figure 3 depicts the Skip-gram architecture model [20].

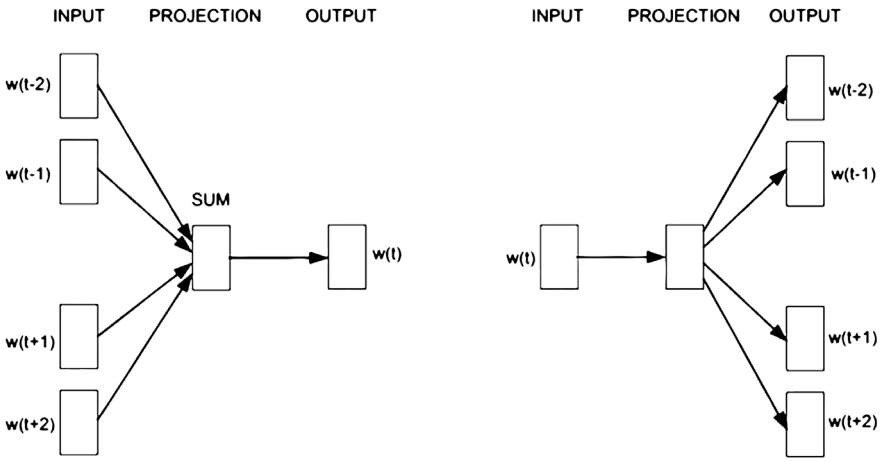

Fig. 3. CBOW and Skip-gram Model Architecture [20]

The skip-gram model maximizes the average log probability which can be seen in equation (4) [21].

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{t+j}|w_t) \quad (4)$$

Where wt is the middle word, $wt+j$ is the word after the middle word, and c is the size of the training context [21]. The parameters used in the Word2Vec model in this study include 'vector_size' which is set at values of 100, 200, and 300 that determine the dimension of the word vector. Parameter 'window'=10, which is used to predict the target word. Parameter 'sg'=1, which indicates the use of the Skip-gram algorithm as it tends to produce more accurate word representations. Parameter 'hs'=0, which suggests the use of negative sampling for training. Parameter 'min_count'=1 to include all words in the vocabulary in training, and parameter 'workers'=4 to enable parallel processing with 4 worker threads, which speeds up the training process. The model was trained for 1000 epochs to ensure thorough learning.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

E. Combination Feature TF-IDF Weighted Word2Vec

This feature combination is a text vector representation that combines Word2Vec features extraction with TF-IDF (Term Frequency-Inverse Document Frequency) features extraction [22]. All the word vectors in the text are combined into a single vector by multiplying the TF-IDF value against the Word2Vec vector. Equation (5) displays the TF-IDF Weighted Word2Vec calculation.

$$\mathrm{V}(d) = \frac{\sum_{w \in W(d)} W(t) \cdot \mathrm{v}(t)}{\sum_{w \in W(d)} W(t)} \tag{5}$$

Based on the previous TF-IDF equation, where $W(t)$ is the TF-IDF value of the word t, $V(d)$ is the representation vector of document d, $W(d)$ is the set of words in document d, the Word2Vec vector for the word t is $\mathrm{v}(t)$. Calculating the TF-IDF value and Word2Vec vector for each word in a document is the first step in performing TF-IDF Weighted Word2Vec. If the word is in the Word2Vec model and the TF-IDF word set, multiply the TF-IDF value by the word's Word2Vec vector. According to TF-IDF, this gives more weight to more important words. The resulting vectors are then summed and normalized by dividing them by the total TF-IDF weight for the document.

F. Learning of SVM Model

Support Vector Machine (SVM) is a subset of supervised learning methods used for classification and regression [23]. First developed by Boser, Guyon, and Vapnik, a Support Vector Machine (SVM) is a combination of existing concepts, including margin, hyperplane, and kernel [24]. Support Vector Machine (SVM) aims to separate two different classes, commonly referred to as a hyperplane. This hyperplane acts as an optimal separation method to group the positive class (1) and the negative class (-1) [25]. On the other hand, the primary objective is to locate the hyperplane with the greatest margin, or distance between the two classes [23].

1) SVM Hyperplane:

SVM is a classification method that aims to find the hyperplane with the largest margin, where the quality of the hyperplane is improved through the maximum value of the margin. Margin, in this context, refers to the distance between the hyperplane and the support vector, i.e. the point closest to the hyperplane [26]. Hyperplane formation in SVM has a structure of two classes, namely -1 and +1, which are separated by a hyperplane. SVM hyperplane formation can be seen in Figure 4.

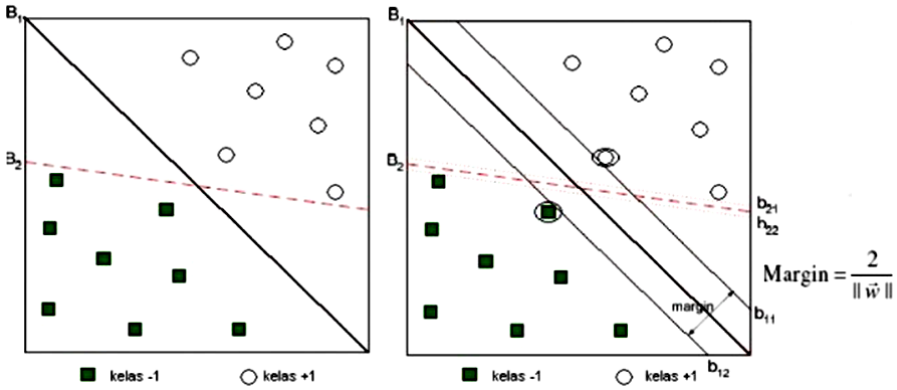

Fig. 4. SVM Hyperplane Formation

To achieve maximum margin, the hyperplane value needs to be determined first, as described in equation (6), in condition (7) [26].

$$\frac{1}{2}\|w\|^2 \tag{6}$$

$$w.x_i + b = 0 \tag{7}$$

The maximum margin can be predicted in equations (8), and (9) [26]. If the vector $\vec{w}$ belongs to the +1 class, it can be explained as follows [24].

$$w.x_i + b \leq -1 \tag{8}$$

Meanwhile, if the vector $\vec{w}$ belongs to class -1, it can be described as follows [24].

$$w.x_i + b \geq +1 \tag{9}$$

2) SVM Classifier:

At the hyperplane prediction stage, we need to apply the hypothesis function *h_0*, which can be identified in equation (10) [9].

$$h_0(x_i) = \begin{cases} +1, & \text{if } w \times x + C \geq 0 \\ -1, & \text{otherwise} \end{cases} \quad (10)$$

Using these two equations, the resulting hyperplane can be classified into class +1 and class -1 [9].

SVM utilizes various commonly used kernel functions, including Linear, Polynomial, Radial Basis Function (RBF), and Sigmoid [9]. These kernel functions play a role in transforming data to higher dimensions to handle dimensionality problems [9]. The four SVM kernel functions can be seen in equations (11), (12), (13), and (14) [18].

a) Linear

Linear kernel is the simplest type of kernel function, which is based on the dot multiplication operation of two vectors [18].

$$K(x_i, x_j) = x_i^T x_j \quad (11)$$

equation (6) shows x_i is the training data and x_j is the test data [9].

b) Polynomial

A polynomial kernel is a type of kernel function with degree *d*, where *d* and *r* are predefined parameters [18].

$$K(x_i, x_j) = (\gamma . X_i^T X_j + r)^d, \gamma > 0 \quad (12)$$

equation (7) shows x_i is the training data, x_j is the test data and d is the polynomial degree [9].

c) Radial Basis Function (RBF)

The Radial Basis Function (RBF) kernel, also known as the Gaussian kernel function, uses parameters to set the distance [18].

$$K(x_i, x_j) = exp(-\gamma |X_i^T X_j|^2), \gamma > 0 \quad (13)$$

d) Sigmoid

Sigmoid kernel, where $tanh(\alpha) = 2\sigma\ (\alpha) - 1$ and $(\alpha) = \frac{1}{(1+exp(\alpha)}$ [18].

$$K(x_i, x_j) = \tanh(\gamma . X_i^T X_j + r) \quad (14)$$

equation (14) shows x_i is the training data, x_j is the test data and r is the coefficient [9].

G. Evaluation

Evaluation is carried out using an evaluation matrix, namely the confusion matrix to measure the classification results to know the level of effectiveness of the model that has been built. Confusion matrix is a method used to evaluate the classification model that has been made to determine the accuracy of the prediction. Table VII outlines the four conditions of confusion matrix accuracy testing.

TABLE VII
CONFUSION MATRIX

Actual Value	Predicted Value	
	Positive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

Four conditions that can be seen in Table 7, namely:

1) True Positive (TP) where the actual value is positive (1), and the predicted value is positive (1).
2) False Positive (FP) where the actual value is negative (0), and the predicted value is positive (1).
3) False Negative (FN) where the actual value is positive (1), and the predicted value is negative (0).
4) True Negative (TN) where the actual value is negative (0), and the predicted value is negative (0).

The performance metrics used consist of Accuracy, Precision, Recall, and F1-Score which can be seen in equations (15), (16), (17), and (18).

1) Accuracy

Accuracy is the percentage of the total number of correct predictions out of the entire data collection, which is determined by equation (14).

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (15)$$

2) Precision

Precision is the ratio of actual positive predictions to overall positive results. The equation for calculating precision can be seen in equation (15).

$$Precision = \frac{TP}{TP + FP} \quad (16)$$

3) Recall

Recall is the ratio of actual positive predictions to all positive data. The equation to calculate recall can be seen in equation (8).

$$Recall = \frac{TP}{TP + FN} \quad (17)$$

4) F1-score

The F1-score is calculated by taking the average value of precision and recall calculating model performance, which can be seen in equation (17).

$$F1 - score = 2\left(\frac{Precision \times Recall}{Precision + Recall}\right) \quad (18)$$

III. RESULT AND DISCUSSION

A. Result

In this study, tests have been carried out that aim to evaluate the most effective and influential feature extraction in the sentiment classification of Nanovest online investment application reviews on the Google Play Store. In labeling, there are conditions where the number of labels in each class is not balanced, the class labeled 1 (positive) is much more than the class labeled 0 (negative) with a ratio of 9.272 for positively labeled data and 1.357 for negatively labeled data as shown in Figure 5.

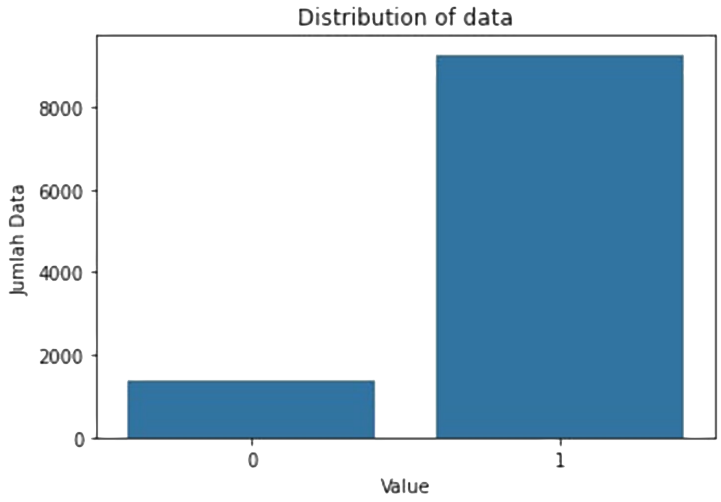

Fig. 5. Distribution of Data

This leads to instability in the constructed model which will only predict one majority class (positive) and ignore the minority class (negative). When the classes are unbalanced, the overall accuracy may look high if the majority class is dominant. However, this can be misleading as the model cannot predict the minority class. Therefore, resampling using Random Oversampling (ROS) is performed to handle the class imbalance by adding more examples from the minority class to the dataset until the number of examples equals the number of examples from the majority class. By equalizing the number of examples between the minority and majority classes, the model is trained on a more balanced data distribution. As a result, the positive and negative classes became balanced with a ratio of 7.417 for positively labeled classes and 7.417 for negatively labeled classes as can be seen in Figure 6.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 10, No. 2, Juni 2025, Pp. 863-874

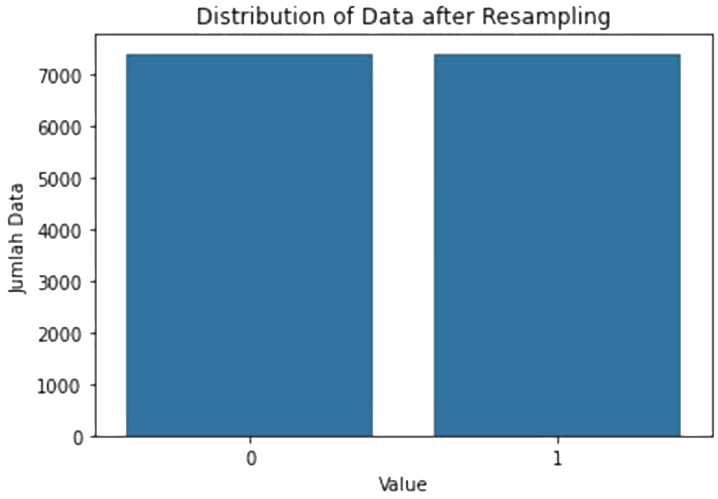

Fig. 6. Distribution of Data After Resampling

This can improve the prediction accuracy of the minority class without compromising the accuracy of the majority class, thus improving the overall performance of the model. This helps the model to learn from the minority data better and reduces the bias towards the majority class. As a result, the model becomes more stable as it can better predict both classes. Although this method is effective for improving minority class accuracy, overfitting can potentially occur due to excessive data addition. Other alternative resampling techniques include Random Undersampling (RUS), which removes some majority class data, however, this may lead to the loss of valuable information. Another method is the Synthetic Minority Oversampling Technique (SMOTE) which creates new minority class instances by interpolating existing instances. These techniques can be selected based on dataset characteristics and research objectives to improve model performance and generalization.

The testing phase uses TF-IDF with 'max_feature' and 'ngram_range' parameters to set or limit the maximum number of features. Max feature is limited to the 20,000 most frequently occurring features in the document, yielding better results within that range. The n-gram parameter is set to 'ngram_range'= (1, 3) so that unigrams, bigrams, and trigrams are considered as features. This helps the model to capture more text context and produce richer and more informative features. Furthermore, the skip-gram model is used in Word2Vec as it tends to produce more accurate word representations than the CBOW model. Tests used vector sizes 100, 200, and 300 to see the balance between accuracy and performance, with 'window'=10 to capture a wide enough context for each word. TF-IDF and Word2Vec are combined into a TF-IDF Weighted Word2Vec combination feature by multiplying the TF-IDF value against the Word2Vec vector, then the resulting vectors are summed and normalized.

The classification algorithm used is Support Vector Machine (SVM) with 'RandomizedSearchCV' parameter optimization to find the best hyperparameters randomly. This process is done by defining the distribution for each parameter to be optimized, such as kernel type, C-value, and gamma. Next, several parameter combinations will be randomly selected from the predefined distributions. During the test, this optimization tests 50 parameter combinations. Each parameter combination is evaluated using cross-validation with 10-fold, and the best performance from the 10-fold average will be selected as the best parameter for the SVM model. RandomizedSearchCV was chosen due to its efficiency in exploring a large parameter space in a faster and more effective way compared to the GridSearchCV method, which requires more time and larger computational resources. The use of RandomizedSearchCV greatly affects the final performance of the model by finding the optimal combination of parameters, which ultimately improves the accuracy and generalization of the model. With optimized parameters, the model can avoid overfitting and underfitting, resulting in more accurate and reliable classifications for data that has never been seen before.

Cross-validation will also be used to evaluate the performance of the SVM model that has been trained, by dividing the dataset into K subsets or folds. One-fold serves as test data for each iteration, and the remainder serves as training data., this process is repeated until each fold has been used as test data once, and the results of each fold are combined to get the average evaluation metric. The results of each fold will have its confusion matrix which will be summed up to get a combined confusion matrix and finally calculate the evaluation metric to get the final accuracy.

B. Discussion

The tests were conducted with trained SVM models and four kernels: linear, polynomial, radial basis function (RBF), and sigmoid. Evaluation metrics are derived from the combined confusion matrix, cross-validation with 10-folds, and the data division is 80% for training data and 20% percent for test data.

TABLE VIII
TF-IDF FEATURE EXTRACTION EVALUATION RESULTS

TF-IDF with SVM	Accuracy	Precision	Recall	F1-score
Linear	90.40%	86.08%	96.38%	90.94%
Poly	88.63%	82.32%	98.39%	89.64%
RBF	91.53%	92.12%	90.83%	91.47%
Sigmoid	93.31%	94.50%	91.97%	93.22%

Based on Table VIII, SVM with TF-IDF achieved the best accuracy of 93.31% on the Sigmoid kernel, this kernel functions like neurons in a neural network, which makes it very flexible in capturing complex non-linear patterns in the features generated by TF-IDF. This flexibility makes it very effective for text classification that has complex feature relationship patterns. The other kernels achieve an accuracy of around 90%, but there is a notable difference in the polynomial kernel with an accuracy of 88.63%, which may be due to the difficulty of the kernel in handling the high dimensionality of TF-IDF features. This kernel can handle non-linear relationships by transforming the feature space to a higher dimension. However, in the case of TF-IDF, the already high dimensionality coupled with polynomial complexity may lead to overfitting. This reduces the ability of the model to generalize to the test data, leading to lower accuracy. The Linear Kernel achieved an accuracy of 90.40%, showing a fairly good performance in linearly separating the data, which reflects its ability to handle TF-IDF features effectively. However, since it is unable to capture more complex non-linear relationships, it has a slightly lower accuracy than the non-linear kernel. On the other hand, the RBF kernel shows better performance with 91.53% accuracy. This kernel utilizes a radial basis function which allows it to separate data in a higher dimensional space, making it more adaptive in handling complex data variations. This makes it more effective in modeling non-linear relationships in TF-IDF features compared to polynomial kernels.

TABLE IX
WORD2VEC FEATURE EXTRACTION EVALUATION RESULTS

Word2Vec with SVM	Vector Size	Accuracy	Precision	Recall	F1-score
Linear	100	85.72%	83.44%	89.22%	86.23%
	200	91.91%	91.84%	91.85%	91.85%
	300	92.75%	93.06%	92.40%	92.73%
Poly	100	94.30%	96.50%	91.91%	94.15%
	200	94.13%	96.72%	91.30%	93.93%
	300	94.55%	96.19%	92.61%	94.36%
RBF	100	95.11%	95.78%	94.58%	95.17%
	200	95.44%	95.94%	94.75%	95.34%
	300	95.46%	95.91%	94.97%	95.44%
Sigmoid	100	93.24%	93.71%	92.70%	93.20%
	200	93.79%	94.27%	93.51%	93.89%
	300	93.71%	93.88%	93.48%	93.68%

Based on Table IX, SVM with Word2Vec achieves the highest accuracy of 95.46% on the RBF kernel with a vector size of 300. The accuracy increases significantly with larger vector sizes from 93.24% at vector size 100 to 95.46% at vector size 300, which indicates that RBF can capture deeper semantic information with more complex feature representations. The linear kernel achieved 85.72% accuracy at vector size 100 and improved at vector size 300 with 92.75% accuracy, but its performance was lower compared to the non-linear kernel due to its nature of only being able to separate data linearly. The Polynomial kernel achieves 94.30% accuracy at vector size 100 and increases to 94.55% at vector size 300. However, its performance is slightly lower than the RBF kernel at the same vector size. In contrast, the Sigmoid kernel has the best performance at a vector size of 200 with an accuracy of 93.79% but slightly decreases when the vector size is increased to 300 with an accuracy of 93.71%. This may be due to its more sensitive nature with larger vector sizes which suggests that larger vector sizes do not always result in improved performance for the sigmoid kernel. The vector size in Word2Vec refers to the dimension of the feature representation for each word in the text. Larger vectors tend to improve SVM model accuracy and metrics by capturing more semantic information through richer, more detailed features. The increased accuracy with larger vector sizes indicates that complex feature representations better discriminate data in higher-dimensional spaces.

TABLE X
TF-IDF WEIGHTED WORD2VEC EVALUATION RESULTS

TF-IDF Weighted Word2Vec with SVM	Accuracy	Precision	Recall	F1-score
Linear	93.54%	93.88%	93.16%	93.52%
Poly	92.73%	90.39%	95.67%	92.95%
RBF	95.29%	95.44%	95.11%	95.27%
Sigmoid	93.92%	94.00%	93.62%	93.81%

Based on Table X, SVM with the combination of TF-IDF Weighted Word2Vec features achieves the highest accuracy of 95.29% on the RBF kernel with a vector size of 300. This combination provides a rich and informative feature representation so that the RBF kernel can effectively handle complex data variations, causing its accuracy to be the highest among all tested kernels. For each kernel, it reaches its best performance with the highest accuracy at a vector size of 300, where the vector is the result of a combination of TF-IDF Weighted Word2Vec features. For example, the Sigmoid Kernel reaches an accuracy of 93.92%. Although the combination of features provides a rich representation, the performance of the sigmoid kernel is not as optimal as the RBF kernel due to its sensitivity to parameter changes. The Linear kernel achieved an accuracy of 93.54%. This kernel can utilize the features well but is not fully optimal as it can only separate the data linearly, which causes its accuracy to be slightly lower than the non-linear kernel. The Polynomial kernel, on the other hand, achieved the lowest accuracy among the other kernels at 92.73%. This kernel can capture more variations in the data. However, this added complexity can lead to overfitting, reducing overall accuracy.

In comparison, this study shows that combining TF-IDF and Word2Vec features with SVM, especially using RBF kernel, results in higher accuracy (95.29%) compared to previous studies. For example, research [10] achieved 89.7% accuracy using Linear kernel and TF-IDF, and research [11] achieved 84.7% accuracy with TF-IDF and SVM. This shows that the combination of TF-IDF and Word2Vec can better capture the statistical and semantic aspects of the text, thus improving the performance of the model. In addition, research [7] using Word2Vec with RBF kernel achieved lower accuracy for Gojek (87%) and Grab (82%) compared to the results of this study. This shows that the integration of TF-IDF with Word2Vec improves the feature richness and the ability of the model to handle complex data variations. However, this study has the potential of overfitting on TF-IDF with high-dimensional data, especially when the dataset is not large enough or unbalanced.

The combination of TF-IDF and Word2Vec features significantly improves the understanding and performance of the model in text classification. However, it may also increase the computational complexity. TF-IDF will look at words that appear frequently and are statistically important. For example, words such as "value," "loss," "profit," and "market" may have a high TF-IDF weight because they are frequently seen in the investment context. This provides the model with information on the focus of the context and helps in identifying keywords that contribute to positive or negative sentiment. On the other hand, Word2Vec provides a vector representation of words that captures the semantic relationships between them. For example, the word "profitable" may often be seen together with words such as "investment," "increase," or "return," whereas the word "loss" may often be seen together with words such as "risk," "decrease," or "loss." Word2Vec can map these words into a multidimensional space vector, which shows their semantic relatedness in the context of investment. This combination of features allows the model to understand not only the frequency and importance of a particular word (TF-IDF) but also the deeper semantic relationships between the words (Word2Vec). Thus, the combination of TF-IDF and Word2Vec helps the model to capture both statistical and semantic aspects of the text, enabling a more comprehensive understanding and more accurate sentiment classification.

This research shows that using a combination of TF-IDF and Word2Vec features optimized with SVM models can improve the quality of sentiment classification in online investment applications. Adopting this method can improve accuracy in analyzing user reviews and comments, enabling better detection of sentiment trends, and providing a more personalized experience through tailored recommendations and prompt notifications. In addition, more accurate sentiment analysis can improve risk management by detecting potential problems or user dissatisfaction and enabling support teams to respond quickly to negative feedback. With deeper insights into user needs and wants, applications can develop new products and services that better suit the market, as well as optimize existing features.

IV. CONCLUSION

This research is motivated by how effective Word2Vec and TF-IDF feature extraction are in sentiment classification on online investment platforms using SVM. The results show that the combination of Word2Vec and TF-IDF Weighted Word2Vec is more effective in improving sentiment classification performance compared to

TF-IDF. Word2Vec with skip-gram model and vector size 300 and window 10 produces the highest accuracy on RBF kernel, which is 95.46%, whereas TF-IDF produces the lowest accuracy on Sigmoid kernel which is 93.31%. Although this accuracy is still quite good, TF-IDF itself may be less effective than Word2Vec because it only considers word frequency and does not capture semantic context. The combination of TF-IDF Weighted Word2Vec on RBF kernel with a vector size of 300 achieved an accuracy of 95.29%, slightly lower than Word2Vec. This may be due to the added complexity of the feature combination. Overall, Word2Vec gives the highest accuracy, followed by the combination of TF-IDF Weighted Word2Vec, and TF-IDF. This research includes improved feature extraction so that future research can explore features or feature combinations using other Word Embedding such as Glove and FastText or other feature extraction, in addition to the use of larger and balanced datasets can provide a broader picture and improve model capabilities.

REFERENCES

[1] J. Mantik *et al.*, "Sentiment Analysis of Online Investment Applications on Google Play Store using Random Forest Algorithm Method," *Jurnal Mantik*, vol. 5, no. 4, pp. 2203–2209, Feb. 2022.

[2] J. V. Girsang, I. K. Jaya, and H. G. Simanullang, "PENERAPAN METODE NAÏVE BAYES CLASSIFIER PADA SENTIMEN ANALISIS APLIKASI INVESTASI KEUANGAN DIGITAL Studi Kasus: Bareksa Dan Bibit," *METHOMIKA Jurnal Manajemen Informatika dan Komputerisasi Akuntansi*, vol. 7, no. 2, pp. 225–230, Oct. 2023, doi: 10.46880/jmika.Vol7No2.pp225-230.

[3] J. Riset and A. Terpadu, "The Public Interest of Tegal in Stock Investment during the Covid-19 Pandemic," *JURNAL RISET AKUNTANSI TERPADU*, vol. 15, no. 1, pp. 1–10, 2022, doi: 10.35448/jrat.v15i1.12479.

[4] A. Dwiki, A. Putra, and S. Juanita, "Analisis Sentimen Pada Ulasan Pengguna Aplikasi Bibit Dan Bareksa Dengan Algoritma KNN," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 8, no. 2, pp. 636–646, Jun. 2021, doi: 10.35957/jatisi.v8i2.962.

[5] L. E. Pradana and Y. Ruldeviyani, "Sentiment Analysis of Nanovest Investment Application Using Naive Bayes Algorithm," *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, vol. 12, no. 2, pp. 283–293, Jul. 2023, doi: 10.23887/janapati.v12i2.62302.

[6] S. Lestari, S. Saepudin, P. Studi, S. Informasi, F. Sains, and D. Teknologi, "Support Vector Machine: Analisis Sentimen Aplikasi Saham di Google Play Store," *JUSIFO JURNAL SISTEM INFORMASI*, vol. 7, no. 2, pp. 81–90, Dec. 2021, doi: 10.19109/jusifo.v7i2.9825.

[7] E. Suryati, A. Ari Aldino, N. Penulis Korespondensi, and E. Suryati Submitted, "Analisis Sentimen Transportasi Online Menggunakan Ekstraksi Fitur Model Word2vec Text Embedding Dan Algoritma Support Vector Machine (SVM)," *JURNAL TEKNOLOGI DAN SISTEM INFORMASI*, vol. 4, no. 1, pp. 96–106, Mar. 2023, doi: 10.33365/jtsi.v4i1.2445.

[8] D. T. Wisudawati, T. iani W. Utami, and P. R. Arum, "Analisis Sentimen Terhadap Dampak Covid-19 Pada Performa Tokopedia Menggunakan Support Vector Machine," *Seminar Nasional Variansi ...*, pp. 87–96, 2020, [Online]. Available: https://ojs.unm.ac.id/variansistatistika/article/view/19508

[9] R. Sitepu, "The Analysis of Support Vector Machine (SVM) on Monthly Covid-19 Case Classification," *International Journal on Information and Communication Technology (IJoICT)*, vol. 8, no. 2, pp. 40–52, Dec. 2022, doi: 10.21108/ijoict.v8i2.671.

[10] F. F. Irfani, "ANALISIS SENTIMEN REVIEW APLIKASI RUANGGURU MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE," *JBMI (Jurnal Bisnis, Manajemen, dan Informatika)*, vol. 16, no. 3, pp. 258–266, Feb. 2020, doi: 10.26487/jbmi.v16i3.8607.

[11] S. Fransiska and A. Irham Gufroni, "Sentiment Analysis Provider by.U on Google Play Store Reviews with TF-IDF and Support Vector Machine (SVM) Method," *Scientific Journal of Informatics*, vol. 7, no. 2, pp. 203–212, Nov. 2020, doi: 10.15294/sji.v7i2.25596.

[12] O. I. Gifari, M. Adha, I. Rifky Hendrawan, F. Freddy, and S. Durrand, "Analisis Sentimen Review Film Menggunakan TF-IDF dan Support Vector Machine," *JIFOTECH (JOURNAL OF INFORMATION TECHNOLOGY*, vol. 2, no. 1, pp. 36–40, Mar. 2022, doi: 10.46229/jifotech.v2i1.330.

[13] A. V. Febrianti, "Analisis Sentimen Data Ulasan Pengunjung Objek Wisata Lawang Sewu Kota Semarang Pada Situs Tripadvisor," pp. 1–101, 2020, [Online]. Available: http://lib.unnes.ac.id/41832/1/4112317002.pdf

[14] Y. Sibaroni, "Perbandingan Pembobotan Fitur TF-IDF dan TF-ABS Dalam Klasifikasi Berita Online Menggunakan Support Vector Machine (SVM)," *e-Proceeding of Engineering*, vol. 10, no. 3, pp. 3652–3663, Jun. 2023.

[15] D. E. Cahyani and I. Patasik, "Performance comparison of tf-idf and word2vec models for emotion text classification," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 5, pp. 2780–2788, Oct. 2021, doi: 10.11591/eei.v10i5.3157.

[16] R. G. Ramli and Y. Sibaroni, "Klasifikasi Topik Twitter menggunakan Metode Random Forest dan Fitur Ekspansi Word2Vec," *e-Proceeding of Engineering*, vol. 9, no. 1, pp. 79–92, Feb. 2022.

[17] H. M. Lee and Y. Sibaroni, "Comparison of IndoBERTweet and Support Vector Machine on Sentiment Analysis of Racing Circuit Construction in Indonesia," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 7, no. 1, pp. 99–106, Jan. 2023, doi: 10.30865/mib.v7i1.5380.

[18] N. Arifin, U. Enri, and N. Sulistiyowati, "PENERAPAN ALGORITMA SUPPORT VECTOR MACHINE (SVM) DENGAN TF-IDF N-GRAM UNTUK TEXT CLASSIFICATION," *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)* , vol. 6, no. 2, pp. 129–136, Dec. 2021.

[19] F. M. Rizky, J. Jondri, and K. M. Lhaksmana, "Twitter Sentiment Analysis of Kanjuruhan Disaster using Word2Vec and Support Vector Machine," *Building of Informatics, Technology and Science (BITS)*, vol. 5, no. 1, pp. 219–227, Jun. 2023, doi: 10.47065/bits.v5i1.3612.

[20] M. Ghifari Adrian, S. Suryani Prasetyowati, and Y. Sibaroni, "Effectiveness of Word Embedding GloVe and Word2Vec within News Detection of Indonesian uUsing LSTM," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 7, no. 3, pp. 1180–1188, Jul. 2023, doi: 10.30865/mib.v7i3.6411.

[21] F. Wahyu Kurniawan and W. Maharani, "Analisis Sentimen Twitter Bahasa Indonesia dengan Word2Vec," *e-Proceeding of Engineering*, vol. 7, no. 2, p. 7821, Aug. 2020, [Online]. Available: https://code.google.com

[22] M. Mohammedid and N. Omar, "Question classification based on Bloom's taxonomy cognitive domain using modified TF-IDF and word2vec," *PLoS One*, vol. 15, no. 3, pp. 1–21, Mar. 2020, doi: 10.1371/journal.pone.0230442.

[23] Naufal Adi Nugroho and Erwin Budi Setiawan, "Implementation Word2Vec for Feature Expansion in Twitter Sentiment Analysis," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 5, pp. 837–842, Oct. 2021, doi: 10.29207/resti.v5i5.3325.

[24] C. Gilang Kencana and Y. Sibaroni, "Klasifikasi Sentiment Analysis pada Review Buku Novel Berbahasa Inggris dengan Menggunakan Metode Support Vector Machine (SVM)," *e-Proceeding of Engineering*, vol. 6, no. 3, pp. 10451–10462, Dec. 2019.

[25] N. R. Robynson and Y. Sibaroni, "Analisis Tren Sentimen Masyarakat Terhadap Pembatasan Sosial Berskala Besar Kota Jakarta Menggunakan Algoritma Support Vector Machine," *e-Proceeding of Engineering*, vol. 8, no. 5, pp. 10166–10178, Oct. 2021.

[26] A. R. Abelard and Y. Sibaroni, "Multi-aspect sentiment analysis on netflix application using latent dirichlet allocation and support vector machine methods," *JURNAL INFOTEL*, vol. 13, no. 3, pp. 128–133, Aug. 2021, doi: 10.20895/infotel.v13i3.670.