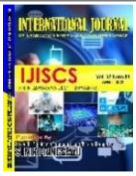

IJISCS (International Journal of Information System and Computer Science)
IJISCS, Volume 7, Number 3, December 2023
E ISSN: 2598-246X; P ISSN: 2598-0793, pp. 200-208
Accredited SINTA 4 Number 79/E/KPT/2023
https://jurnal.ftikomibn.ac.id/index.php/ijiscs

GOOGLE PLAY STORE USERS COMMENT REVIEW CLASSIFICATION USING SVM CLASSIFIER AND RANDOM FOREST

Muhammad Rafi Hadiyasa[1], Sani Muhamad Isa[2]
[1,2]Computer Science Department, BINUS Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia
[1,2]Jl. KH. Syahdan No. 9, Kemanggisan, Palmerah Jakarta, Indonesia

*Corresponding author
muhammad.hadiyasa@binus.ac.id[1]*
sani.m.isa@binus.ac.id[2]

Article history:
Received August 17, 2023
Revised September 22, 2023
Accepted October 12, 2023

Keywords:
Multiclass;
SVM Classifier;
Text Classification;
Random Forest;

Abstract
In today's digital age, social media stands as a dynamic arena where individuals freely express their thoughts and opinions, from succinct tweets on Twitter to expansive narratives on platforms like Facebook and Instagram. However, amidst this vast sea of user-generated content, a glaring void persists a definitive rating system capable of distilling the nuanced sentiments embedded within these diverse commentaries. This study thus emerges as a pioneering endeavor, poised to bridge this crucial gap in sentiment analysis. Leveraging the transformative potential of the Word2vec methodology in the preprocessing phase, researchers embark on a comprehensive journey to classify comments on a meticulous 1-5 rating scale, thereby unraveling the multifaceted spectrum of sentiments encapsulated within them. Complementing this groundbreaking approach, the Random Forest classification model is harnessed to bolster the analytical prowess of the study. The resultant accuracy score of 60.4% stands as a testament to the study's significant strides towards achieving a deeper understanding of comment sentiment in the realm of social media. However, this is merely the inception of a promising trajectory; the study's findings hold the promise of not only refining sentiment analysis techniques but also revolutionizing diverse sectors, from market research to product development. With this study, the narrative of sentiment analysis transcends the confines of academia, beckoning forth a new era of nuanced comprehension and meaningful engagement within the sphere of social media commentary. As the study concludes, it leaves behind a compelling call to action, inviting further exploration and innovation in this dynamic field.

I. INTRODUCTION

In the rapidly evolving landscape of technology, advancements have permeated various facets of daily life, with Indonesia standing as a vivid example of this transformative wave. The

Google Play store provides users with access to a wide array of third-party applications for download and use [1]. Among the pioneering services shaping this technological revolution is the Google Play Store, an offering from the industry giant Google, providing a platform where users not only download applications but also actively engage in a feedback-driven discourse.

Within the realm of the Google Play Store, users wield the power to articulate their perspectives through a spectrum of means, ranging from textual comments to precise star ratings on a calibrated scale of one to five. This nuanced rating system signifies a spectrum of user satisfaction, where a solitary star represents the nadir of contentment, while five stars denote the zenith of approval. Collectively, these evaluations coalesce to furnish a comprehensive portrait of the application, subsequently cataloged based on the aggregate ratings furnished by the user community.

The significance of this user-generated feedback within the Google Play Store ecosystem cannot be overstated. It serves as a crucial compass for developers, guiding them towards refining existing applications and inspiring the development of future iterations. This kind of dialogue enables developers to offer potential solutions for the concerns raised in the review or to request users to elaborate on their dissatisfaction with the app [2]. The forthcoming research endeavor seeks to build upon precedent studies, particularly one that harnessed the potential of the Random Forest approach [3]. This antecedent research centered on leveraging comments data gleaned from the Gojek application, sourced from the Google Play Store, to glean valuable insights from user ratings.

In the pursuit of an even more granular analysis, this study endeavors to harness comments data as a bedrock for rating classification, with the dual aim of enhancing the efficacy of prior research endeavors. To accomplish this, we propose the application of Word2vec for robust preprocessing, followed by the strategic deployment of the Random Forest algorithm for the classification of the extensive corpus of comments harvested from the Google Play Store. Through this methodological tapestry, our intent is to engender a substantial contribution to the ongoing discourse surrounding user feedback analysis and its pivotal role in refining and advancing applications within the Google Play Store ecosystem.

II. RELATED WORKS

Comment Classification have been developed in recent years. As in several related studies that have been collected, it can be seen through the methods used in the research. On research [4], the authors developed a multiclass text classification model using Naive Bayes classifier. Naive Bayes has demonstrated its effectiveness in various real-world scenarios, such as text classification, medical diagnosis, and performance management of system [5], [6]. The model is used to classify text in the Chinese language into different categories such as sport, tourism, automobile, etc. The model uses Vector Space Model to represent the text document as vectors. The result of the text classification model shows that it obtained the highest F1 score of 0.96 in the automobile category. The authors stated that the low F1 scores in other categories might be the result of some data which belong to multiple categories, and this issue might be addressed by the authors in future studies.

On research [7], the authors were interested to compare how well do different word embedding architectures perform when utilized for text classification. The architectures used are Glove, Word2vec, and Fasttext. The reason for implementing these architectures in text classification is due to the fact that the generated word embeddings can capture semantic and contextual relationships between words. The random forest classifier is trained using the word embeddings generated by each architecture. It is then tested to classify the tweets based on four emotion classes: happy, surprised, angry, and sad. Each classifier in the random forest ensemble is created using an independently sampled random vector from the input vector[8]. The study results show that the best performing model is the model with the parameters of 200 estimators in Random Forest classifier and a Fasttext word embedding with 300 dimensions. This model achieved the highest precision score of 91% and provides a satisfactory result for using word embeddings in text classification.

This study [9]aims to demonstrate the effectiveness of sentiment analysis in identifying customer opinions on an online shopping platform, specifically flipkart.com. In this research, the data will encompass customer reviews regarding the products. The system will navigate through the URLs to locate these opinions. This process involves extracting data from the internet to ensure precision in accordance with user specifications. The information acquired from the website will then be carefully examined to isolate the reviews for subsequent analysis and processing. Sentiment analysis is performed using Natural Language Processing (NLP). Opinion mining relies on Natural Language Processing (NLP) to extract the underlying meaning of opinionated words and sentences [10]. And subsequently, VADER, which is a component of the NLTK module, can be utilized. VADER is a rule-based model for general sentiment analysis that was evaluated in the study. It was compared to 11 established benchmarks, including ANEW, LIWC, General Inquirer, Senti WordNet, as well as machine learning approaches that use Naive Bayes, Maximum Entropy, and Support Vector Machine (SVM) algorithms [11]. VADER employs a lexicon of words to identify negative and positive sentiments. It also evaluates the sentiments to determine sentiment scores. In essence, VADER aids in assessing whether a text exhibits positive, negative, or neutral sentiment based on the words it contains.

In [12], the authors were able to build word vectors based on sentiment lexicons for sentiment predictions in the Greek and English languages. Sentiment lexicons are used to give sentiment information when training the word vector model using Word2vec method. The proposed hybrid vectorization process, which combines lexicon-based features and word-embedding approaches, provides both the semantic and contextual relationships between words as well as the sentiment orientation of a document. The word vector model is then used in training the classifier model which uses Support Vector Machine (SVM) algorithm. The result shows that the proposed methodology does not surpass the state-of-the-art in the English language in terms of accuracy. It provides an accuracy improvement of 5.7% compared to the lexicon-based representations, and only 1.6% accuracy improvement compared to plain Word2vec representation. However, it gives high accuracy for the Greek language and is state-of-the-art in which it gives an accuracy improvement of 5.5% and 10.2% compared to lexicon-based representation and Word2vec representation respectively.

Similar to the previously mentioned study, in [13], the authors also used Word2vec to generate word embeddings and applied it in text classification using SVM classifier. This study [14] aims to classify user interests based on the comments they posted on the social media platform Twitter. Comments are classified into 5 categories according to their topics which are sports, travel, fashion, food, and religion. The pre-labelled tweets used in this study are originally collected and published in[15]. After the word embeddings have been generated, it will then be inputted into Convolutional Neural Network architecture for deep features extraction. SVM classifier with linear kernel is then used to predict the sentiment of each tweet. The result of the proposed model shows that it achieved the highest accuracy of 97.3%.

After reading and studying on a related researchm the steps that will be used on conducting the research this time will be similar to what the previous research has done. With a method that is expected to have a better output than the methods that have been applied previously. The method that will be used for the classification is Random Forest, while in the preprocessing stage, we will use Word2vec.

III. METHODOLOGY

The overarching objective of this study is to delve into a comprehensive classification of comments provided by users of a specific application, meticulously sourced from the Google Play Store. Within this ambit, a nuanced evaluation spanning five distinct categories is envisaged. These categories are delineated by the familiar 1 to 5-star rating system, where a solitary star denotes a less-than-ideal experience, and conversely, a five-star rating signifies a high level of user satisfaction. This multifaceted assessment is poised to furnish invaluable insights into user sentiment and engagement.

In the pursuit of assembling a robust dataset for analysis, the research team employed a sophisticated web scraping technique. This methodological approach was instrumental in extracting user-generated comments from the sprawling expanse of the Google Play Store.

Specifically, the comments were sourced directly from the play.google.com website, where a treasure trove of authentic, unfiltered user feedback resides.

To facilitate this data retrieval process, the researchers leveraged the versatility and power of the Python programming language. A custom-designed script was meticulously crafted to automate the task of comment extraction. This script exhibited a remarkable capability to seamlessly navigate through the website, deftly capturing comments with precision and efficiency.

The automation aspect of the script was especially pivotal, as it not only expedited the data collection process but also ensured a comprehensive coverage of comments across the platform. By dynamically scrolling through the website, the script adeptly captured a diverse array of comments, reflecting a broad spectrum of user experiences and sentiments. Following the successful extraction of data in CSV format, a critical phase in the research process ensued-data preprocessing. This pivotal step was undertaken to refine the dataset, ensuring it aligns seamlessly with the researchers' specific focus on comments expressed in the Indonesian language.

3.1. Preprocessing

In this study, a comprehensive preprocessing approach was employed to enhance the quality of the textual data under examination. The first step involved case folding, which entailed converting all uppercase characters to their lowercase counterparts. This adjustment is crucial for ensuring uniformity in the text, as it eliminates potential discrepancies caused by variations in capitalization. Given that the comments originate from users, it was observed that a substantial portion contained non-alphanumeric characters, including emoticons. These elements were subsequently removed, along with any extraneous punctuation marks, to streamline the text for further analysis.

Following the case folding and character cleansing process, stemming was applied as the next preprocessing stage. Stemming involves transforming words into their base or root form, thereby reducing inflected words to their fundamental linguistic stem. This aids in consolidating the data and reducing redundancy, as it ensures that words with similar meanings are represented consistently. Additionally, stemming facilitates a more efficient analysis of the text, as it allows for the recognition of common themes and trends across various forms of a word.

To further refine the data, a stopword removal procedure was implemented. Stopwords are commonly occurring words that do not contribute significant meaning to a sentence, such as conjunctions and prepositions. By removing these uninformative elements, the focus shifts towards the more substantive content within the comments. This step is pivotal in enhancing the relevance of the data for subsequent analysis, as it refines the corpus to contain only words that carry substantive meaning.

While the preprocessing tasks largely relied on the Natural Language Toolkit (NLTK) library, it is important to note that due to the comments being in Indonesian, an Indonesian-specific library was utilized. This decision was instrumental in ensuring the accuracy and appropriateness of the preprocessing techniques applied. One of the notable challenges encountered during this process was the prevalence of slangs in the Indonesian language. As a result, a considerable effort was directed towards identifying and eliminating these colloquial expressions using the stopword preprocessing. This step was essential in order to maintain the integrity and reliability of the data for subsequent analyses.

Table 1. Example of preprocessing comments

Original Comment	Aplikasinya aneh sih, kok tidak mau login. Gimana mau pake :(
Comment Cleansing	aplikasinya aneh sih kok tidak mau login gimana mau pake
Stopword Removal	aplikasi aneh tidak mau login gimana mau
Stemming	aplikasi aneh tidak mau login gimana mau
Tokenization	[aplikasi], [aneh], [tidak], [mau], [login], [gimana], [mau]

3.2. Word embedding using Word2vec

At this stage, the previously processed data is used in the word embedding process. Word2vec that will be used aims to embed words using the library from Gensim. There this process takes data from the preprocess in the form of tokenized text as input and produces a model as output. this model is a vocabulary with a vector for each word. Using the library from Gensim with the function "most_similar" will provide the closest 10 words to the input word. In this case, the word "bagus" will return "mantap" because the function of "most_similar" will give the same result as the input word.

```
a = "bagus"
model2.wv.most_similar(positive=a)
```

```
[("'bagus", 0.8152381181716919),
 ('bngat', 0.7150043845176697),
 ('bintanglima', 0.7098819613456726),
 ('mudsah', 0.6978936195373535),
 ('bangetttt', 0.6967946290969849),
 ('pertahankan', 0.6891748905181885),
 ('brguna', 0.6880638599395752),
 ('woow', 0.6801451444625854),
 ('mantaab', 0.6797555685043335),
 ('bagus.', 0.6785733699798584)]
```

Figure 1. Word2vec Most Similar Result

The provided code in Figure 1 of using Word2Vec with the Gensim library. In this code snippet, the variable a is assigned the value "good". Subsequently, the model2 is employed to find words that are most similar to the word "bagus". The function model2.wv.most_similar(positive=a) returns a list of words that are most similar to the word "good" based on their embedding vectors. In this context, "most_similar" entails finding words whose embedding vectors are similar or close to the embedding vector of the word "bagus".

The result of this call will be a list of words along with their similarity scores to the word "bagus". It signifies that the word "bagus" has a similarity score of 0.8 with the word "bagus", "bngat " has a similarity score of 0.75, and so forth. A higher similarity score indicates a greater resemblance between the words. This explanation can be included in your paper to clarify the code snippet.

3.3. Random Forest Algorithm

In classifying after the embedding process, by conducting a training process on the classification model. One of the classification algorithms that will be used in this study is the Random Forest Algorithm. Comments that have been obtained from Google Play Store will be trained with the labels that have been owned by each comment.

The classification process is divided into 5 categories according to the rating given by the user, which is between 1 and 5 stars. 1 star which means the worst, and 5 star which means the most satisfying. After the training process is carried out on the classification model, the model is then tested using the Scikit-learn prdeiction function. The testing process provides results in the form of accuracy, precision, recall, and F1-score in figure 2.

	precision	recall	f1-score	support
1	0.72	0.27	0.39	173
3	0.39	0.62	0.48	189
5	0.66	0.64	0.65	193
accuracy			0.52	555
macro avg	0.59	0.51	0.50	555
weighted avg	0.58	0.52	0.51	555

Figure 2. Result

IV. RESULT AND ANALYSIS

The data that we collected was 16.361 comments from the Google Play Store. The data for each rating can be seen in Figure 3. In picture 3, we gathered comments with a rating of 5, totaling 9627 comments, followed by a rating of 1 with 2765 comments, 4-star ratings with 2498 comments, 3-star ratings with 924 comments, and the least with a rating of 2, which had 546 comments.

```
df2.rating.value_counts()
```

```
5    9627
1    2765
4    2498
3     924
2     546
Name: rating, dtype: int64
```

Figure 3. Data from each rating

The vector model has 100 dimensions and is trained using Word2vec with 15 iterations. The vector represents each word that has been found. The following results from the accuracy obtained in Figure 4. In the section within Figure 4, there is also Python code. This code imports the balance accuracy_score function from the sklearn.metrics library and then utilizes it to assess the model's accuracy. y_test represents the actual labels of the test data, while prediction_SVM denotes the predictions generated by the model using the Support Vector Machine (SVM) algorithm. The outcome of this accuracy measurement is then multiplied by 100.

```
from sklearn.metrics import balanced_accuracy_score
balanced_accuracy_score(y_test, predictions_SVM)*100
```

```
60.48346351540018
```

```
balanced_accuracy_score(y_test, predictions_log)*100
```

```
56.62390897148851
```

Figure 4. Accuracy

Researchers are also trying to get the accuracy of several other methods as shown in Figure 4. In Figure 4, researchers are exploring the accuracy of various methods. Specifically, the code snippet balance_accuracy_score(y_test, prediction_log)*100 within Figure 4 calculates the balance accuracy score for a method known as prediction_log. In this context, y_test represents the actual labels of the test data, while prediction_log denotes the predictions generated by a particular method, potentially a logistic regression model. The balance_accuracy_score function is employed to evaluate accuracy, taking into account the balance between different classes. The resultant accuracy score is then multiplied by 100, likely to present it as a percentage.

```
SVM Accuracy Score ->  58.198198198198206
SVM precision Score ->  56.89397122139859
SVM recall Score ->  58.198198198198206


SVM balanced Accuracy Score ->  60.0
SVM balanced precision Score ->  59.52363080477192
SVM balanced recall Score ->  60.0


logistic Accuracy Score ->  56.75675675675676
logistic precision Score ->  54.298316523325106
logistic recall Score ->  56.75675675675676


logistic balanced Accuracy Score ->  56.03603603603604
logistic balanced precision Score ->  53.57986499482594
logistic balanced recall Score ->  56.03603603603604


SVM2 Accuracy Score ->  56.93693693693693
SVM2 precision Score ->  55.781073031574856
SVM2 recall Score ->  56.93693693693693


SVM2 balanced Accuracy Score ->  56.93693693693693
SVM2 balanced precision Score ->  55.633589277370646
SVM2 balanced recall Score ->  56.93693693693693


etree Accuracy Score ->  72.25225225225225
etree precision Score ->  71.98925950598488
etree recall Score ->  72.25225225225225


etree balanced Accuracy Score ->  72.43243243243244
etree balanced precision Score ->  73.60357767200426
etree balanced recall Score ->  72.43243243243244
```

Figure 5. The accuracy of various methods.

In Figure 5. The performance evaluation results of various classification methods (SVM, Logistic Regression, SVM2, and Decision Tree) indicate different levels of accuracy, precision, and recall. Accuracy measures how well the model correctly predicts, while precision indicates how accurate the positive predictions of the model are. On the other hand, recall measures how well the model can identify all positive cases.

The first SVM method shows an accuracy of around 58.19%, precision of 56.89%, and recall of 58.19%. The SVM method with imbalanced class handling (Balance) performs slightly better with an accuracy of 60.0%, precision of 59.2%, and recall of 60.0%. It can be concluded that SVM with imbalanced class handling tends to yield better results in terms of accuracy and recall.

Logistic Regression, on the other hand, has an accuracy of approximately 57.76%, precision of 54.29%, and recall of 56.75%. With imbalanced class handling, its performance slightly decreases with an accuracy of 56.03%, precision of 53.57%, and recall of 56.03%. This method generally performs similarly, whether with or without imbalanced class handling.

The SVM2 method shows an accuracy of about 56.36%, precision of 55.78%, and recall of 56.93%. Meanwhile, SVM2 with imbalanced class handling results in an accuracy of 56.93%, precision of 55.63%, and recall of 56.93%. In other words, imbalanced class handling on SVM2 provides a slight improvement in precision performance, but it does not significantly affect accuracy and recall.

Lastly, the Decision Tree method (etree) exhibits the best results with an accuracy of around 72.25%, precision of 71.98%, and recall of 72.25%. With imbalanced class handling, the performance improves slightly with an accuracy of 72.43%, precision of 73.60%, and recall of 72.43%. Among all evaluated methods, Decision Tree provides the best results in terms of accuracy, precision, and recall.

When choosing the most suitable method, it is important to consider the specific goals and needs of the classification problem at hand. For example, if the main focus is on high accuracy, Decision Tree might be the best choice. However, if precision or recall holds special significance in the context of the problem, the method choice can be tailored to those needs.

V. CONCLUSION

In the course of this study, a comprehensive approach leveraging the Word2vec preprocessing method was employed to enhance the quality of the textual data. This encompassed critical tasks such as stopword removal, case folding, tokenization, and data cleansing. The classification of comments was conducted using the Random Forest algorithm, with additional methods like Support Vector Machine (SVM) being employed for comparative analysis. Notably, among these methods, the Etree balance method emerged as the standout performer, boasting impressive scores of 72.4% for accuracy, 73.6% for precision, and 72.4% for recall. The model developed in this study demonstrates commendable proficiency in predicting and categorizing user comments sourced from the Google Play Store platform. This capacity holds substantial implications for leveraging sentiment analysis in the refinement and optimization of products and services offered to users. As the study lays the groundwork for future research, there exists a multitude of avenues for exploration and refinement. Specifically, in the context of the Indonesian language, addressing the prevalence of slang presents a critical challenge. Given that slang often deviates from conventional linguistic norms, incorporating specialized techniques or integrating slang-specific vocabulary into the analysis process could be pivotal in enhancing accuracy. This targeted approach acknowledges the linguistic intricacies of Indonesian, thereby ensuring a more adept and reliable sentiment analysis framework. Furthermore, ongoing research endeavors could delve into advanced machine learning techniques or alternative preprocessing methods to further augment the accuracy and efficacy of sentiment classification. By continuously refining the existing methodologies, the research community can foster the development of a sophisticated sentiment analysis model, tailored precisely for Indonesian language contexts. This iterative process of enhancement and optimization will undoubtedly contribute to a more comprehensive and precise framework, with far-reaching applications in diverse industries and sectors.

REFERENCES

[1] N. Viennot, E. Garcia, and J. Nieh, “A measurement study of Google Play,” in *Performance Evaluation Review*, Association for Computing Machinery, Jun. 2014, pp. 221–234. doi: 10.1145/2591971.2592003.

[2] S. Hassan, C. Tantithamthavorn, C. P. Bezemer, and A. E. Hassan, “Studying the dialogue between users and developers of free apps in the Google Play Store,” *Empir Softw Eng*, vol. 23, no. 3, pp. 1275–1312, Jun. 2018, doi: 10.1007/s10664-017-9538-9.

[3] R. I. Kurnia, Y. D. Tangkuman, and A. S. Girsang, “Classification of user comment using word2vec and SVM classifier,” *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 1, pp. 643–648, 2020, doi: 10.30534/ijatcse/2020/90912020.

[4] J. J. C. Mca, M. Phil, and D. Ph, “GREEN – MOBILE CLOUD COMPUTING AND ITS APPLICATIONS 3G / 4G,” vol. 3, no. 20, pp. 342–344, 2016.

[5] P. Domingos and M. Pazzani, “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss,” Kluwer Academic Publishers, 1997.

[6] I. Rish, “An empirical study of the naive Bayes classifier.”

[7] 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*, no. January 2019, pp. 90–95, 2019, doi: 10.1109/IALP.2018.8629262.

[8] M. Pal, “Random forest classifier for remote sensing classification,” *Int J Remote Sens*, vol. 26, no. 1, pp. 217–222, Jan. 2005, doi: 10.1080/01431160412331269698.

[9] Amity University and Institute of Electrical and Electronics Engineers, *10th International Conference on Cloud Computing, Data Science & Engineering : proceedings of the Confluence 2020 : 29-31 January 2020, Amity University, Uttar Pradesh, India.*

[10] M. T. Khan, M. Durrani, A. Ali, I. Inayat, S. Khalid, and K. H. Khan, “Sentiment analysis and the complex natural language,” *Complex Adaptive Systems Modeling*, vol. 4, no. 1. Springer, Dec. 01, 2016. doi: 10.1186/s40294-016-0016-9.

[11] S.-I. Ao and International Association of Engineers, *International MultiConference of Engineers and Computer Scientists : IMECS 2019 : 13-15 March, 2019 : The Royal Garden Hotel, Kowloon, Hong Kong*.
[12] M. Giatsoglou, M. G. Vozalis, K. Diamantaras, A. Vakali, G. Sarigiannidis, and K. C. Chatzisavvas, "Sentiment analysis leveraging emotions and word embeddings," *Expert Syst Appl*, vol. 69, pp. 214–224, 2017, doi: 10.1016/j.eswa.2016.10.043.
[13] A. H. Ombabi, W. Ouarda, and A. M. Alimi, "Deep learning CNN–LSTM framework for Arabic sentiment analysis using textual information shared in social networks," *Soc Netw Anal Min*, vol. 10, no. 1, pp. 1–13, 2020, doi: 10.1007/s13278-020-00668-1.
[14] Kalasalingam Academy of Research and Education. IEEE Student Branch., Institute of Electrical and Electronics Engineers, and IEEE Power & Energy Society, *IEEE International Conference on Intelligent Techniques in Control, Optimization & Signal Processing : INCOS-'19 : 11th-13th April 2019.*
[15] B. Y. Pratama and R. Sarno, "Personality classification based on Twitter text using Naive Bayes, KNN and SVM," *Proceedings of 2015 International Conference on Data and Software Engineering, ICODSE 2015*, pp. 170–174, 2016, doi: 10.1109/ICODSE.2015.7436992.