Enhancing Semi-Supervised Sentiment Analysis Through Hyperparameter Tuning Within Iterations: A Comparative Study Using Grid Search and Random Search

Agus Sasmito Aribowo *, Nur Heri Cahyana, and Yuli Fauziah

Department of Informatics, Universitas Pembangunan Nasional Veteran Yogyakarta, Yogyakarta 55281, Indonesia
sasmito.skom@upnyk.ac.id

Abstract. Improper placement of hyperparameter tuning in the semi-supervised sentiment analysis process can potentially decrease processing speed and fail to enhance performance. This research explores the impact of hyperparameter tuning within iterations of semi-supervised sentiment analysis. Two architectural approaches are tested: one with hyperparameter tuning at the beginning and another with tuning at each iteration. Grid search and random search are employed for hyperparameter tuning. The study demonstrates that hyperparameter tuning within iterations enhances the performance of semi-supervised sentiment analysis models. The experiments conducted on four diverse datasets demonstrated that hyperparameter tuning within iterations generally leads to improved performance. Model B, which applies hyperparameter tuning within iterations, showed better accuracy, precision, recall, and F1-score than Model A, which conducts tuning at the beginning. Additionally, grid search outperformed random search, although the differences in performance were not highly significant, approximately from 0.1% to 2% in all experiments. These results suggest that incorporating hyperparameter tuning within the iterations of semi-supervised sentiment analysis can enhance model performance, and grid search can be a more effective method for this task, especially when time efficiency is a priority. The choice between grid and random search depends on the trade-off between time and performance. Future research can extend these findings to different machine learning techniques and datasets.

Keywords: Hyperparameter Tuning, Semi-supervised Learning, Sentiment Analysis.

1 Introduction

1.1 Research Background

Sentiment analysis, also known as sentiment analysis or opinion mining, is a natural language processing technique used to identify, extract, and analyze sentiments, opin-

A. Putro Suryotomo and H. Cahya Rustamaji (eds.), *Proceedings of the 2023 1st International Conference on Advanced Informatics and Intelligent Information Systems (ICAI3S 2023)*,
Advances in Intelligent Systems Research 181,
https://doi.org/10.2991/978-94-6463-366-5_23

ions, or emotions contained in text. The main goal of sentiment analysis is to determine whether a text has a positive, negative, or neutral sentiment [1]. Sentiment analysis techniques are part of natural language processing, statistics, and machine learning to classify text into appropriate sentiment categories [2].

Semi-supervised sentiment analysis is an innovative approach that combines elements of supervised learning and unsupervised learning to analyze sentiment and opinions in text [3]. In this context, "semi-supervised" means that the model uses data that has been labeled based on its sentiment (supervised) and data that does not have a sentiment label (unsupervised). This methodology has several advantages compared to conventional sentiment analysis, especially by utilizing various data sources and providing a more comprehensive understanding of sentiment patterns. By leveraging labeled and unlabeled data, semi-supervised sentiment analysis not only maximizes data utilization but also reduces the time and costs associated with manual labeling of large datasets, providing an efficient and cost-effective solution [2]. This inclusive approach improves model performance by identifying subtle sentiment patterns that may be difficult to capture with labeled data alone. Additionally, the flexibility of semi-supervised sentiment analysis means it can adapt to changing data environments and sentiment trends over time. Although this approach offers many advantages, it requires careful data management and good model design to optimize its effectiveness. It emphasizes the importance of high-quality data and thoughtful model design in its implementation.

Hyperparameter tuning in the context of sentiment analysis is an essential aspect in the development of machine learning models for classifying sentiment or opinion in text. This involves adjusting model parameters not taught by the model, known as hyperparameters. In sentiment analysis, the main goal is to ensure that the model can recognize and correctly classify sentiment in text, which can often be a complex task due to complex human language and variations in how people express opinions. Performing effective hyperparameter tuning for a machine learning classifier can lead to a substantial increase in accuracy [4].

Grid search and random search are techniques in hyperparameter tuning to find the optimal parameter for a machine learning model. Grid search is a systematic approach that involves searching for combinations of hyperparameters by testing all possible combinations of one or more hyperparameters. The advantage of grid search is that it tries all combinations to use the best hyperparameter combination. However, this can also be time-consuming if there are many hyperparameters or values to test [5]. Random search is a more flexible and random approach. It tests hyperparameter combinations by randomly selecting hyperparameter values within a specified range. The advantage of random search is that it can save time because it does not test all combinations [4]. Additionally, it is possible to find good hyperparameter combinations more quickly if done randomly. Thus, hyperparameter tuning becomes an important element in optimizing the performance of sentiment analysis models and ensuring their adaptability to various types of text and changes in an ever-changing data environment. In conventional sentiment analysis models, hyperparameter tuning can be applied before modelling so that the model formed already uses parameters called the best machine learning parameters (or best parameters). The problem is where hyperparameter tuning is applied in semi-supervised sentiment analysis, considering that some types of semi-supervised sentiment analysis not only work in one iteration but can even be done in dozens of iterations until they reach ideal conditions.

According to Al Laith, et.al. in their semi-supervised learning (SSL) model called ArasenCorpus [6] and in the semi-supervised text analysis (SSTA) model from previous research [7], [8], [9], it requires several iterations to reach a convergent condition or a stopping condition because all the data has been labeled. Each iteration will create a new machine learning model using the new pseudo-labeled dataset to annotate the remaining unlabeled dataset. So, in what position should the hyperparameter tuning be placed to get the new best parameters? So, in these SSL steps, if hyperparameter tuning is applied, several questions arise:

1. Are the best parameters only sought at the beginning of the SSL process, namely in the initial model after it has been formed using a labeled dataset at the beginning of the iteration? Then, the best parameters are applied to the next iterations. So, in this case, the hyperparameters are placed before the SSL process.
2. Are the best parameters calculated and applied in each iteration using a dataset with pseudo labels with the consequence that it will involve a hyperparameter tuning process in each iteration, and of course, it will reduce performance (slow)? So, in this case, hyperparameter tuning is placed together with machine learning for each SSL iteration. This has been studied in the Semi-supervised Learning model Using Logistic Regression for Sentiment Annotation in previous research [10].
3. Can the random search method perform the same as grid search after applying hyperparameter tuning at each SSL iteration?

As an initial hypothesis, of course, logically applying hyperparameter tuning at each iteration will provide more advantages because the best parameters will be measured and redetermined for each iteration. If the increase is not significant, the first method can be chosen for several conditions, especially for a faster SSL process. This research also aims to find out how good the difference in improvement is.

1.2 Literature Review

This literature review will look at how hyperparameter tuning is applied in SSL, what method is used for the hyperparameter tuning process, where generally existing research uses grid search or random search.

Research on SSL for sentiment analysis includes [11], which uses the grid search approach for hyperparameter tuning. This research performed a grid-search strategy using different hyperparameters for each classification model in SSL. The experiments in research [7] were carried out at several threshold figures, between 1% and 40%. Hyperparameter tuning is placed before the SSL process is executed [11] .The next research is a new semi-supervised sentiment analysis from documents uses Semi-supervised Distributed Bag of Word (DBOW) algorithm in the training procedure section. The research places a hyperparameter tuning process at each algorithm iteration [12]. The next research is the development of unsupervised (domain-independent) and semi-supervised (domain-specific) methods for Turkish. This research applies hyperparameter tuning using the scikit-learn framework for several machine learning algorithms. Hyperparameter tuning is determined before the SSL process is run [13]. The next research is about the three-step semi-supervised hybrid approach model for aspect-based sentiment analysis. This research experimented with various values of the hyperparameters, and the best performance of the model was successfully obtained for generating the vocabulary of each class. Each SSL iteration uses the same parameter value (K) [14]. The next research analyses the effectiveness of several semi-supervised learning approaches for opinion spam classification. In the research, the hyperparameters were obtained iteratively by testing different parameters to find the optimal outcome using grid search. In the self-training and co-training, only the hyperparameters of the base estimators, such as in SVM, NB, and RF, are optimized. So, this research involves updating the hyperparameter tuning process at each iteration of the SSL algorithm [15]. More complete and specific research on hyperparameters is research about comparative analysis of many hyperparameter tuning techniques. This research applies and compares several hyperparameter tuning methods, namely Grid Search, Random Search, Bayesian Optimization, Particle Swarm Optimization (PSO), and Genetic Algorithm (GA). They are used to optimize the accuracy of six machine learning algorithms. Based on observations of the SSL algorithm included in this paper, it is known that this research involves updating the hyperparameter tuning process at each iteration of the SSL algorithm [16]. Another research is a semi-supervised learning model for text annotation. The SSL employed a Support Vector Machine and a Random Forest algorithm and used grid and random search to tune the Support Vector Machine and Random Forest parameters. This research involves updating the hyperparameter tuning process at each iteration of the SSL algorithm [5]. Based on the literature review, several studies place hyperparameter tuning at the beginning of SSL, and some apply it in each iteration of the SSL algorithm. However, it has yet to be discovered which one has higher performance and is more efficient in processing time.

2 Research Method

The research steps include six stages, starting from obtaining labeled public datasets for experiments, data preparation including data cleaning, vectorization with TF-IDF, dividing experimental data into training data, and testing data. The experimental process uses two SSLs with different architectures at the location of hyperparameter tuning and continues with discussion and conclusions.

2.1 Datasets

This research focuses on sentiment analysis in Indonesia. There are 4 (four) datasets used for testing in this research (**Table 1**). The dataset will be pre-processed, namely data cleaning, removing non-alphabetic words, converting to lowercase, removing stop words, removing slang words, and stemming. After the data is clean, continue with vectorization using TF-IDF.

Table 1. Dataset used in this study.

Code	Dataset name	Record number	Classes
HS	Indonesian Hate Speech	13168	2 classes: (Hate Speech, Non Hate Speech)
SR	Sentiment Ridife	10805	3 classes: (Positive, Negative, Neutral)
EI	Emotion IndoNLU	7079	5 classes: (joy, love, sad, fear, anger, neutral)
SI	Sentiment IndoNLU	12759	3 classes: (Positive, Negative, Neutral)

These datasets in **Table 1** are used to test two types of SSL with different hyperparameter tuning settings: at the beginning of SSL and others at each SSL iteration. We chose the datasets based on the diversity of the number of data classes and the diversity of the data instances. Each dataset will be divided into labeled data and test data with a ratio of 2:8 randomly. The test data will have its labels removed and function as an unlabelled dataset. Labeled data will be divided into 2, namely training and testing data, with a ratio of 8:2. The distribution of the dataset is in **Fig. 1**.

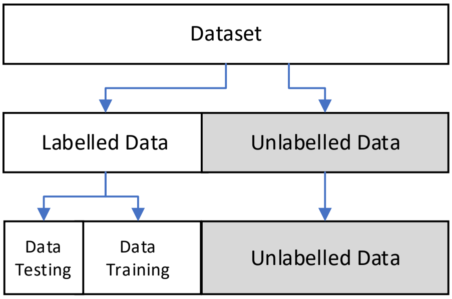

Fig. 1. A distribution of training, testing, and unlabelled data.

Testing data is used to measure model performance after reading training data and hyperparameter tuning. This stage is referred to as the baseline condition assessment stage. Then, the testing data will be used to measure model performance after reading the training data, plus pseudo-labeled data and hyperparameter tuning. This stage is referred to as the final condition assessment stage.

2.2 SSL Model Architecture

SSL Model for the first trial (Model A). The architecture of model A (**Error! Reference source not found.**) require training data, testing data, and unlabelled data. SVM use the training data to build the model. The model will be processed by hyperparameter tuning (leftmost gray box in **Error! Reference source not found.**). The best parameters from hyperparameter tuning will be stored and used in each subsequent iteration. The model will provide automatic annotations on unlabeled data. Weight probability will accompany each new label in the data. The confidence value is set at a threshold of 70%. If the pseudo-label weight exceeds this confidence value, the dataset will become part of the training data. If not, it will be passed to the next iteration as unlabeled data. In the next iteration, SVM will read the training data, which has been added to the dataset with pseudo-labels. SVM will use the same parameters as the SVM model in the previous stage. The model will read unlabelled data (which has been reduced in number) and will provide annotations again. The process continues until no data can reach the threshold value or all unlabelled data has been annotated.

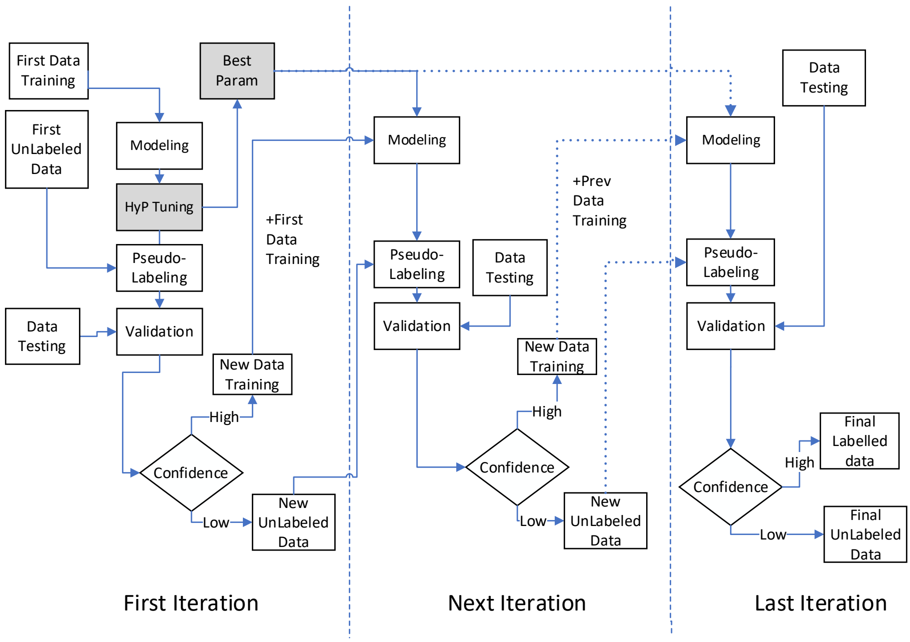

Fig. 2. Architecture of Model A. Hyperparameter tuning only in first iteration

Hyperparameter tuning on Model A repeated using grid search and random search methods. The results of both will also be compared with the SSL process without hyperparameter tuning to see whether hyperparameter tuning improves model performance.

SSL for the second test (Model B). SSL for the second type of hyperparameter testing (model B) is in **Error! Reference source not found.**. Like the first experiment, the Model B in the first iteration need training data, testing data, and unlabelled data. SVM use the training data to build the model. The model is also optimized by hyperparameter tuning process (gray box in First Iteration). There is no storage for best parameter, the optimized model will directly perform sentiment annotation on unlabeled data. In the next iteration, SVM read the training data, which has been added to the dataset with pseudo-labels. The SVM model will be processed by hyperparameter tuning again and get new best parameters (gray box in Next Iteration). Model B will read back the unlabelled data and provide sentiment annotations. The process continues until no data can reach the threshold value of 70% or all data has been successfully annotated.

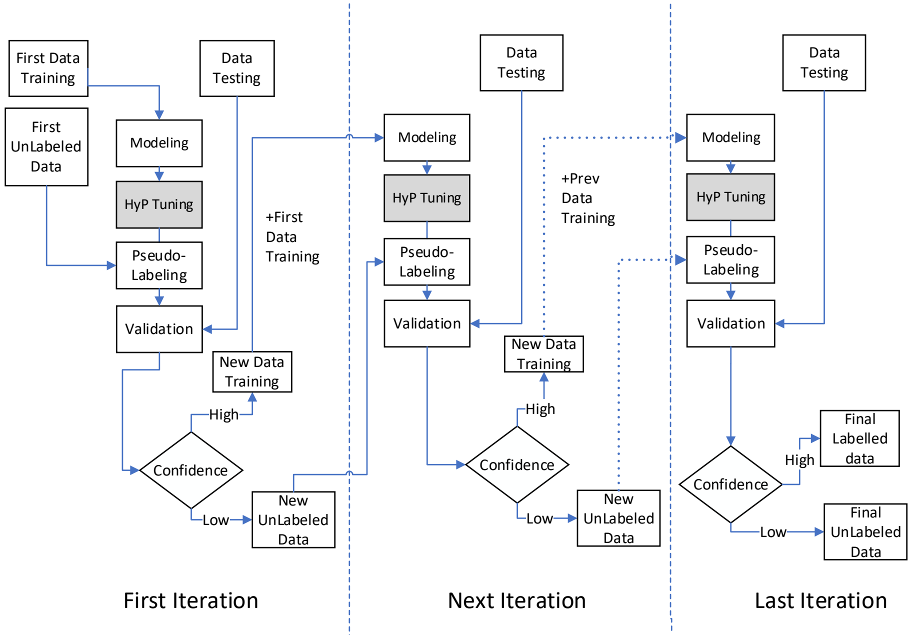

Fig. 3. Architecture of Model B. Hyperparameter tuning located in every iteration.

Like the Model A, hyperparameter tuning on Model B also repeated using grid search and random search methods. The results of both will also be compared with the SSL process without hyperparameter tuning to see whether hyperparameter tuning improves model performance.

2.3 Machine Learning.

SVM (Support Vector Machine) was chosen as the classification algorithm. In this research, a linear kernel SVM is employed, aiming to identify the optimal separation function (hyperplane) that distinguishes opinion data between two binary classes, typically positive and negative sentiments. SVM effectively establishes the separation function between Class 1 and Class 2, creating a clear and maximally wide gap between them [17]. SVM determines the optimal hyperplane by identifying the outermost data points from both classes, situated at the border, and subsequently, the optimal hyperplane is calculated, taking these outermost data points into account. SVM can manage noisy data where there is overlap within certain classes [18].

2.4 Performance Measurement

Confusion matrix provides more insight not only the performance of a predictive model but also into which classes are being predicted correctly or incorrectly and the types of error made. The simplest confusion matrix is in **Table 2**, for a two-class classification problem, with negative class and positive class.

Table 2. Confusion matrix.

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

True Positives (TP) are the classification results that correctly predict positive values, which means that the value of the actual class is true and the value of the predicted class is also true. For example, if the actual class value indicates that this comment is, the positive opinion and predicted class tell the same thing.

True Negatives (TN) are the classification result that correctly predicted negative values, which mean that the value of the actual class is false.

False Positives (FP) – When the actual class is false, and the predicted class is true, if the actual class says this comment is not a positive opinion, but the predicted class says that this comment is a positive opinion.

False Negatives (FN) – When the actual class is true, but the predicted class is false, if the actual class value indicates that this comment is a fanatic opinion, and the predicted class tells that the comment is not a positive opinion.

Accuracy, Precision, Recall and F1 Score. The four parameters in Confusion Matrix calculate accuracy, precision, recall and F1 score. accuracy is a ratio of correctly predicted observation to the total observations. Accuracy is a great measure but only for symmetric datasets which values of false positive and false negatives are almost same.

$$\mathbf{Accuracy = (TP+TN) / (TP+FP+FN+TN)} \quad (1)$$

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate.

$$\mathbf{Precision = TP/TP+FP} \quad (2)$$

Recall (Sensitivity) - recall is the ratio of correctly predicted true observations to all observations in actual class – true.

$$\mathbf{Recall = TP/TP+FN} \quad (3)$$

F1 Score is the weighted average of precision and recall. F1 is usually more useful than accuracy, especially if the class distribution is uneven. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it is better to look at both precision and recall.

$$\textbf{F1 Score} = 2*(\textbf{Recall} * \textbf{Precision}) / (\textbf{Recall} + \textbf{Precision}) \quad (4)$$

3 Research result

3.1 Test Results on HS Dataset.

The HS dataset contains 13168 instances and is divided into two polarities, namely Hate Speech and Non-Hate Speech. The results of SSL performance testing on model A and model B are in **Table 3**.

Table 3. Result of testing on the HS dataset.

Metode	Model	Accuracy	Precision	Recall	F1-Score
Without Hyperparameter		0.784	0.800	0.784	0.788
Grid Search	A	0.806	0.806	0.806	0.806
	B	0.822	0.832	0.817	0.815
Random Search	A	0.796	0.807	0.796	0.799
	B	0.802	0.812	0.805	0.815

3.2 Testing on SR Dataset.

The SR dataset contains 10805 instances and is divided into three polarities, namely positive, negative, and neutral (ternary). The results of SSL performance testing in experiment 1 and experiment 2 are in **Table 4**.

Table 4. Result of testing on the SR dataset.

Metode	Model	Accuracy	Precision	Recall	F1-Score
Without Hyperparameter		0.559	0.686	0.559	0.593
Grid Search	A	0.571	0.710	0.572	0.608
	B	0.595	0.723	0.583	0.622
Random Search	A	0.576	0.659	0.576	0.599
	B	0.581	0.711	0.582	0.608

Test results on the HS and SR dataset (**Table 3** and **Table 4**) show that experiments using Model B generally provide higher performance than experiments using Model A, both using grid search and random search, but not significantly, approximately from 0.1% to 2% (highlighted). Experiments using grid search gave better performance results in all four units of measurement (accuracy, precision, recall, and F1-score) than using random search. This is because grid search tries all possible parameter combinations. The difference in performance between the two is also not very significant. If a faster process is desired, then Model A and random search strategies are good choices.

3.3 Testing on the EI Dataset.

The EI dataset contains 7079 instances and is divided into six polarities, namely joy, love, sad, fear, anger, and neutral. The results of SSL performance testing in experiment 1 and experiment 2 are in **Table 5**.

Table 5. Result of testing on the EI dataset.

Metode	Model	Accuracy	Precision	Recall	F1-Score
Without Hyperparameter		0.662	0.688	0.662	0.657
Grid Search	A	0.702	0.712	0.710	0.695
	B	0.716	0.721	0.716	0.715
Random Search	A	0.695	0.689	0.692	0.685
	B	0.710	0.716	0.710	0.708

3.4 Testing on the SI Dataset.

The SI dataset contains 12759 instances and is divided into three polarities, namely Positive, Negative, and Neutral. The results of SSL performance testing in experiment 1 and experiment 2 are in **Table 6**.

Table 6. Result of testing on the SI dataset.

Metode	Model	Accuracy	Precision	Recall	F1-Score
Without Hyperparameter		0.835	0.861	0.835	0.843
Grid Search	A	0.847	0.877	0.842	0.847
	B	0.852	0.886	0.862	0.851
Random Search	A	0.841	0.867	0.849	0.847
	B	0.848	0.878	0.852	0.849

Based on **Table 5** and **Table 6**, the results obtained from the experiments conducted on the EI and SI dataset also demonstrate a consistent trend where Model B tends to outperform Model A, whether grid search or random search is applied. Nevertheless, the disparity in performance is not significantly substantial, approximately from 0.1% to 2%. More precisely, the grid search experiments produce better outcomes across all the measurement metrics, including accuracy, precision, recall, and F1-score, compared to those utilizing random search. This is also due to grid search tries all possible parameter combinations. The discernible difference in performance between the two methods is also relatively modest. Hence, if a more expedient processing approach is preferred, Model A, in conjunction with random search, is a viable choice.

4 Conclusion

This study explores the impact of hyperparameter tuning in semi-supervised sentiment analysis iterations. Two architectural approaches were tested: one with hyperparameter tuning at the beginning and one with tuning at each iteration. This study implements grid search and random search methods for hyperparameter tuning. The results show that hyperparameter tuning in iterations improves the performance of the semi-supervised sentiment analysis model. Experiments conducted on four different datasets generally show improved performance.

Model B, which applies hyperparameter tuning in iterations, shows better accuracy, precision, recall, and F1 score than Model A, which performs tuning in the initial step. Additionally, grid search outperforms random search, although the performance difference is insignificant, approximately from 0.1% to 2% in all experiments. Therefore, this study shows that incorporating hyperparameter tuning in semi-supervised sentiment analysis iterations can improve model performance, and grid tracing can be a more effective method for this task, especially when time efficiency is a priority. The choice between grid search and random search depends on the trade-off between time and performance. Future research could extend these findings to various machine learning techniques and datasets.

References

1. R. Wijayanti and A. Arisal, "Automatic Indonesian Sentiment Lexicon Curation with Sentiment Valence Tuning for Social Media Sentiment Analysis," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 20, no. 1, 2021, doi: 10.1145/3425632.
2. A. H. Khan and M. Zubair, "Classification of multi-lingual tweets, into multi-class model using Naïve Bayes and semi-supervised learning," *Multimedia Tools and Applications*, vol. 79, no. 43–44, pp. 32749–32767, 2020, doi: 10.1007/s11042-020-09512-2.
3. V. L. Shan Lee, K. H. Gan, T. P. Tan, and R. Abdullah, "Semi-supervised Learning for Sentiment Classification Using Small Number of Labeled Data," *Procedia Computer Science*, vol. 161, pp. 577–584, 2019, doi: 10.1016/j.procs.2019.11.159.
4. E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis," *Informatics*, vol. 8, no. 4, pp. 1–21, 2021, doi: 10.3390/informatics8040079.
5. S. Khomsah, N. H. Cahyana, and A. S. Aribowo, "Hyperparameter Tuning of Semi-Supervised Learning for Indonesian Text Annotation," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 9, pp. 250–256, 2023, doi: 10.14569/IJACSA.2023.0140927.
6. A. Al-Laith, M. Shahbaz, H. F. Alaskar, and A. Rehmat, "AraSenCorpus: A Semi-Supervised Approach for Sentiment Annotation of a Large Arabic Text Corpus," *Applied Sciences (Switzerland)*, vol. 11, no. 5, pp. 1–19, 2021, doi: 10.3390/app11052434.

7. A. S. Aribowo, H. Basiron, and N. F. A. Yusof, "Semi-supervised learning for sentiment classification with ensemble multi-classifier approach," *International Journal of Advances in Intelligent Informatics*, vol. 8, no. 3, pp. 349–361, 2022, doi: 10.26555/ijain.v8i3.929.
8. N. H. Cahyana, S. Saifullah, Y. Fauziah, A. S. Aribowo, and R. Drezewski, "Semi-supervised Text Annotation for Hate Speech Detection using K-Nearest Neighbors and Term Frequency-Inverse Document Frequency," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 10, pp. 147–151, 2022, doi: 10.14569/IJACSA.2022.0131020.
9. W. Wisnalmawati, A. S. Aribowo, and Y. Herawati, "Semi-supervised Learning Models for Sentiment Analysis on Marketplace Dataset," *International Journal of Artificial Intelligence & Robotics (IJAIR)*, vol. 4, no. 2, pp. 78–85, 2022, doi: 10.25139/ijair.v4i2.5267.
10. S. Khomsah and A. S. Aribowo, "Model Semi-supervised Learning Menggunakan Logistic Regression Untuk Anotasi Sentimen," *Open Access Ledger*, vol. 1, no. 4, pp. 171–178, 2022.
11. H. B. B. B and M. das G. V. Nunes, "Semi-supervised Sentiment Annotationof Large Corpora," *Computational Processing of the Portuguese Language*, pp. 385–395, 2018, doi: 10.1007/978-3-319-99722-3.
12. S. Park, J. Lee, and K. Kim, "Semi-supervised distributed representations of documents for sentiment analysis," *Neural Networks*, vol. 119, pp. 139–150, 2019, doi: 10.1016/j.neunet.2019.08.001.
13. C. R. Aydln and T. Güngör, "Sentiment analysis in Turkish: Supervised, semi-supervised, and unsupervised techniques," *Natural Language Engineering*, vol. 27, no. 4, pp. 455–483, 2021, doi: 10.1017/S1351324920000200.
14. A. Kumar, P. Gupta, R. Balan, L. B. M. Neti, and A. Malapati, "BERT Based Semi-Supervised Hybrid Approach for Aspect and Sentiment Classification," *Neural Processing Letters*, vol. 53, no. 6, pp. 4207–4224, 2021, doi: 10.1007/s11063-021-10596-6.
15. A. Ligthart, C. Catal, and B. Tekinerdogan, "Analyzing the effectiveness of semi-supervised learning approaches for opinion spam classification," *Applied Soft Computing*, vol. 101, 2021, doi: 10.1016/j.asoc.2020.107023.
16. E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis," *Informatics*, vol. 8, no. 4, pp. 1–21, 2021, doi: 10.3390/informatics8040079.
17. A. A. Aziz, A. Starkey, and M. C. Bannerman, "Evaluating Cross Domain Sentiment Analysis using Supervised Machine Learning Techniques," in *2017 Intelligent Systems Conference, IntelliSys 2017*, 2017, no. September, pp. 689–696, doi: 10.1109/IntelliSys.2017.8324369.
18. C. CORTES and V. VAPNIK, "Support-Vector Networks," *Machine Learning*, vol. 20, pp. 273–297, 1995, doi: 10.1023/A:1022627411411.