Semi-supervised Learning for Sentiment Classification with Ensemble Multi-classifier Approach

Agus Sasmito Aribowo [a,b,1,*], Halizah Basiron [b,2], Noor Fazilla Abd Yusof [b,3]

[a] Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka, Jalan Hang Tuah Jaya Durian Tunggal, 76100 Melaka, Malaysia
[b] Informatics Department, Universitas Pembangunan Nasional "Veteran" Yogyakarta, Jl. SWK 104 Sleman, Yogyakarta 55283, Indonesia
[1] sasmito.skom@upnyk.ac.id *; [2] halizah@utem.edu.my; [3] elle@utem.edu.my
* corresponding author

ARTICLE INFO

Article history
Received
Revised
Accepted

Keywords
Ensemble Multi-classifier
Semi-supervised
Sentiment Analysis
SVM
Random Forest

ABSTRACT

Supervised sentiment analysis ideally uses a fully labeled data set for modeling. However, this ideal condition requires a struggle in the label annotation process. Semi-supervised learning (SSL) has emerged as a promising method to avoid time-consuming and expensive data labeling without reducing model performance. However, the research on SSL is still limited and its performance needs to be improved. Thus, this study aims to create a new SSL-Model for sentiment analysis. The Ensemble Classifier SSL model for sentiment classification is introduced. The research went through pre-processing, vectorization, and feature extraction using TF-IDF and n-grams. Support Vector Machine (SVM) or Random Forest for tokenization was used to separate unigram, bigram, and trigram in model generation. Then, the outputs of these models were combined using stacking ensemble approach. Accuracy and F1-score were used for the evaluation. IMDB datasets and US Airlines were used to test the new SSL models. The conclusion is that the sentiment annotation accuracy is highly dependent on the suitability of the dataset with the machine learning algorithm. In IMDB dataset, which consists of two classes, it is better to use SVM. In the US Airlines consisting of three classes, SVM is better at improving the model performance against the baseline, but RF is better at achieving the baseline performance even though it fails to maintain the model performance.

1. Introduction

Sentiment analysis is the process of recognizing the writer's positive or negative feelings in documents. Sentiment analysis can be divided into document level, sentence level, and aspect level [1]. Sentiment analysis at the document/sentence level classifies either the positive or the negative sentiments in a document/sentence. Sentiment analysis has been applied to several domains using various techniques. Most supervised sentiment analysis uses machine learning that requires a labeled dataset to train the model. Building a fully labeled dataset takes a lot of effort and cost in obtaining labels for instances [2]. Semi-supervised learning (SSL) has emerged as a promising method to annotate unlabelled data [3]. The semi-supervised approach builds the model from labeled data and incrementally improves the performance of the model by labeling the sentiment polarity of unlabeled instances. This approach avoids time-consuming and expensive data labeling without reducing model performance.

This study aims to create a semi-supervised learning model (SSL-Model) for sentiment analysis using ensemble approach. For vectorization, Term Frequency-Inverse Document Frequency (TF-IDF) and n-gram were applied. The ensemble stacking mechanism was implemented. There were six models set up from two machine learnings (RF and SVM) and three types of vector data (bigram, trigram, and unigram). The combination of TF-IDF with Random Forest performed well in supervised sentiment analysis [4] [5]. The contribution is that a new model uses a combination of TF/IDF, n-gram and SVM or RF can improve SSL labeling accuracy compared to human labeling (baseline), especially in the two datasets.

Various types of semi-supervised learning provide better accuracy in research [8] and [9]. ArasenCorpus is one of the study about a semi-supervised framework to annotate a large Arabic text corpus using small manually annotated tweets. ArasenCorpus study improves the sentiment classification results from 80.37% to 87.4% on SemEval 2017 dataset and from 79.77% to 85.2% on ASTD dataset. ArasenCorpus study has also improved sentiment classification result from 64.10% to 68.1% on ASTD dataset [9], but there is no Arasencorpus research for datasets in English. The next semi-supervised study from [10] has proposed a semi-automatic approach to annotate the Saudi dialect tweets dataset and achieved classifier accuracy of 83% by the Naïve Bayes. Alqarafi et al have suggested a semi-supervised for annotating sentiment corpus for Saudi dialect using Twitter. Their research reach best model with Naive Bayes algorithm, achieved accuracy up to 91% [11]. However the model in [11] has not been tested for datasets in English. Harby et al have determined a semi-supervised for sentiment classification of dialectal reviews with the presence of Modern Standard Arabic (MSA). Harby et al used Support Vector Machines (SVM), Naïve Bayes (NB), Random Forest, and K-Nearest Neighbor (K-NN). This study results that the highest classification accuracy is using SVM algorithm with 92.3% [12]. Carvalho et al in [13] have a prospective experiment to produce a corpus with automated annotation in Brazilian Portuguese. In their study, the Linear SVM presented the best accuracy on the cross-validation, with 0.5533 against 0.5507 from Multinomial Naïve Bayes (the second-best). However the model in [12] and [13] also has not been tested for datasets in English. In English textual review, Balakrishnan et al proposed semi-supervised research for sentiment and emotion analysis using the Support Vector Machine, Random Forest, and Naïve Bayes. In their research, Random Forest gives the best results for sentiment (F1 score = 73.8%) and SVM with F1-Score result of 72,2% [14]. For SSL using SVM in English documents, it has been published in [15] with F1-Score result reach 79,039% on B-SVM (SVM model without SSL) and 79.95% on SSSVM (SVM model with bootstrapping). The performance of both SSL methods still needs to be improved.

This research continues the self-learning mechanism to automate annotations and reduce human dependency. SSL-Model annotates unlabeled datasets using labeled datasets and proceed in several iterations. The first iteration is called the baseline, the classifier model is formed using a manually annotated dataset. The final condition is that all unannotated datasets have been annotated, or the maximum iteration limit has been reached. The focusin this study is in comparing the performance of the baseline condition with the final condition. However, the achievement of the SSL Model is when the final performance does not decrease compared to the baseline. This paper contains: Section 1 presenting an introduction and related works, Section 2 describing the research methodology, Section 3 involving experimental steps and a discussion of experimental results, and Section 4 enclosing conclusions and future research steps.

2. Method

2.1. Data Preprocessing

The US Airlines dataset and the IMDB dataset were used for data processing. These datasets are often used in sentiment analysis model comparison. US Airlines have been investigated in [16], [17], and [18] and IMDB in [5], [19], and [20]. US Airlines consists of 14640 airline reviews downloaded from Kaggle and released by CrowdFlower in CSV format. The US Airlines dataset consists of three classes (positive, neutral, and negative). The IMDB dataset consists of 50,000 documents downloaded from Kaggle at https://www.kaggle.com/code/rafetcan/sentiment-analysis/data. The IMDB dataset consists of two classes which are positive and negative. US Airlines and IMDB needed to be processed first because they were unstructured, and contained non-alphabetical or special characters. Pre-processing through several stages is described in Table 1. This is an example of pre-processing the sentence: “@VirginAmerica you know what would be amazingly awesome? BOSS-FLL PLEASE!!!!!! I want to fly with only you”.

Table 1: Pre-processing stages

Step No.	**Pre-Processing**	
	Method	**Example**
1	Remove Number	@VirginAmerica you know what would be amazingly awesome? BOS-FLL PLEASE!!!!!!! I want to fly with only you.
2	Remove Punctuation	@VirginAmerica you know what would be amazingly awesome BOS-FLL PLEASE I want to fly with only you.
3	Remove Non-Alphabetic Character	VirginAmerica you know what would be amazingly awesome BOS FLL PLEASE I want to fly with only you.
4	Remove Stopword	VirginAmerica amazingly awesome BOS FLL PLEASE want fly only.
5	Convert to Lowercase	virginamerica amazingly awesome bos fll please want fly only.
6	Stemming	virginamerica amazing awesome bos fll please want fly only.
7	Tokenization(unigram, bigram,trigram)	**Unigram :** virginamerica, amazing, awesome, bos, fll, please, want, fly, only. **Bigram :** virginamerica amazing, amazing awesome, awesome bos, bos fll, fll please …. **Trigram :** virginamerica amazing awesome, amazing awesome bos, awesome bos fll,..

After going through the pre-processing stage, the process on documents consisting of at least 2 syllables was continued. There were 14096 US Airline documents that could be continued to the vectorization stage. For the IMDB dataset, all documents could be proceeded to the vectorization stage.

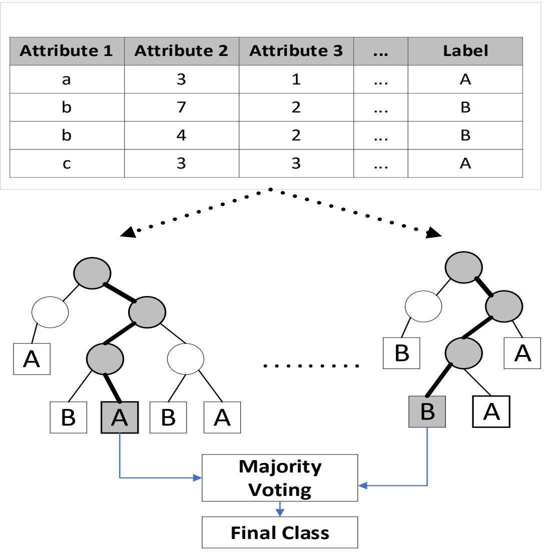
2.2. Vectorization

Term Frequency-Inverse Document Frequency (TF-IDF) is known as an algorithm to calculate the weight of each word in a set of documents. Term frequency is the frequency of occurrence of term Y in document X divided by the total term in document X [21]. IDF reduces the weight of a term if its occurrence is spread throughout the document. TF-IDF vector data is a sparse matrix with dimensions (*n_samples, n_feature*). N_feature is the number of features which is usually the top terms with the largest TF-IDF score. The number of documents is devided by the number of row of dataset becomes n_samples. In very large documents, the features form a very large dimensional matrix because each word that appears in the document is represented by its score [22]. TF-IDF vectorizer has good performance for sentiment analysis in research [23] and [24].

2.3. Modeling

Random Forest (RF) is used to build the ensemble multi-classifier model. Random Forest is an ensemble of decision trees, where the formation of a tree arrangement in a decision tree uses the entropy approach or the Gini index [25]. RF reduces the occurence of overfitting by creating many trees, bootstrapping technique, and splitting nodes. RF split the node using the best split strategy at every node (Fig 1). The final classification is the majority class of these trees. Random Forest has a good performance for sentiment analysis as revealed in research [25] and [26]. In this research, the parameter of Random Forest was set using number of estimators=100, criterion using gini index, and minimum samples split=2.

SVM is also a popular technique for classification. This technique is to find the most optimum hyperplane to split documents from different classes (Fig 2). The SVM strategy to get the optimum hyperplane is to detect the outermost data in the two classes, then find the optimum hyperplane considering the outer data [28]. SVM has a good performance in research [29] and [30]. This study, SVM with kernel parameter Radial Basis Function (RBF) was implemented.



Attribute 1	Attribute 2	Attribute 3	...	Label
a	3	1	...	A
b	7	2	...	B
b	4	2	...	B
c	3	3	...	A

Fig 1. Random Forest

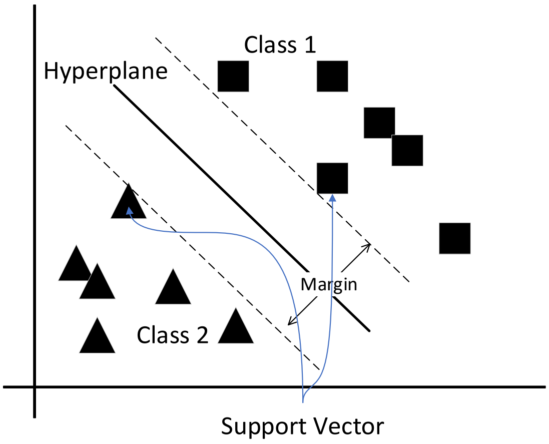


Fig 2. Support Vector Machine

2.4. Architecture

SSL-model architecture was proposed in this study (Fig 3). The process began with reading the annotated input data as data training, data testing, and unlabelled data (the gray boxes in Fig 3). The training data was processed using TF-IDF. The results of TF-IDF vectorization are three vectors: unigram, bigram, and trigram tokenization vector. The three vectors were used to create three models using RF (and SVM as a comparison in the next experiment). The performance of the three models was measured using the F1 score in test 1. This F1 score was used as a weight in the voting process at the threshold calculation stage.

In Fig 3, the result is three models working separately to annotate unlabeled data. Every model produced pseudo labels. Threshold numbers were used to select whether the annotated data (pseudo-labels) was worthy of being training data. Several threshold numbers ranging from 0.6 to 0.9 in the preliminary study had been tried, and 0.6 was used as the threshold number because it produced a more accurate and larger set of labeled documents. The high confidence document would be added to the Data Training. The document with the low confident label would be re-labeled in the next iteration. Iterations in the SSL model ran ten times or until the Unlabeled Data ran out. The output of the model was data training (DT) which had been labeled by humans and machines. The resulting training data was formed into a new classifier model and tested using the F1 score and accuracy in Test 2. Test 2 was a performance measurement of the SSL Model.

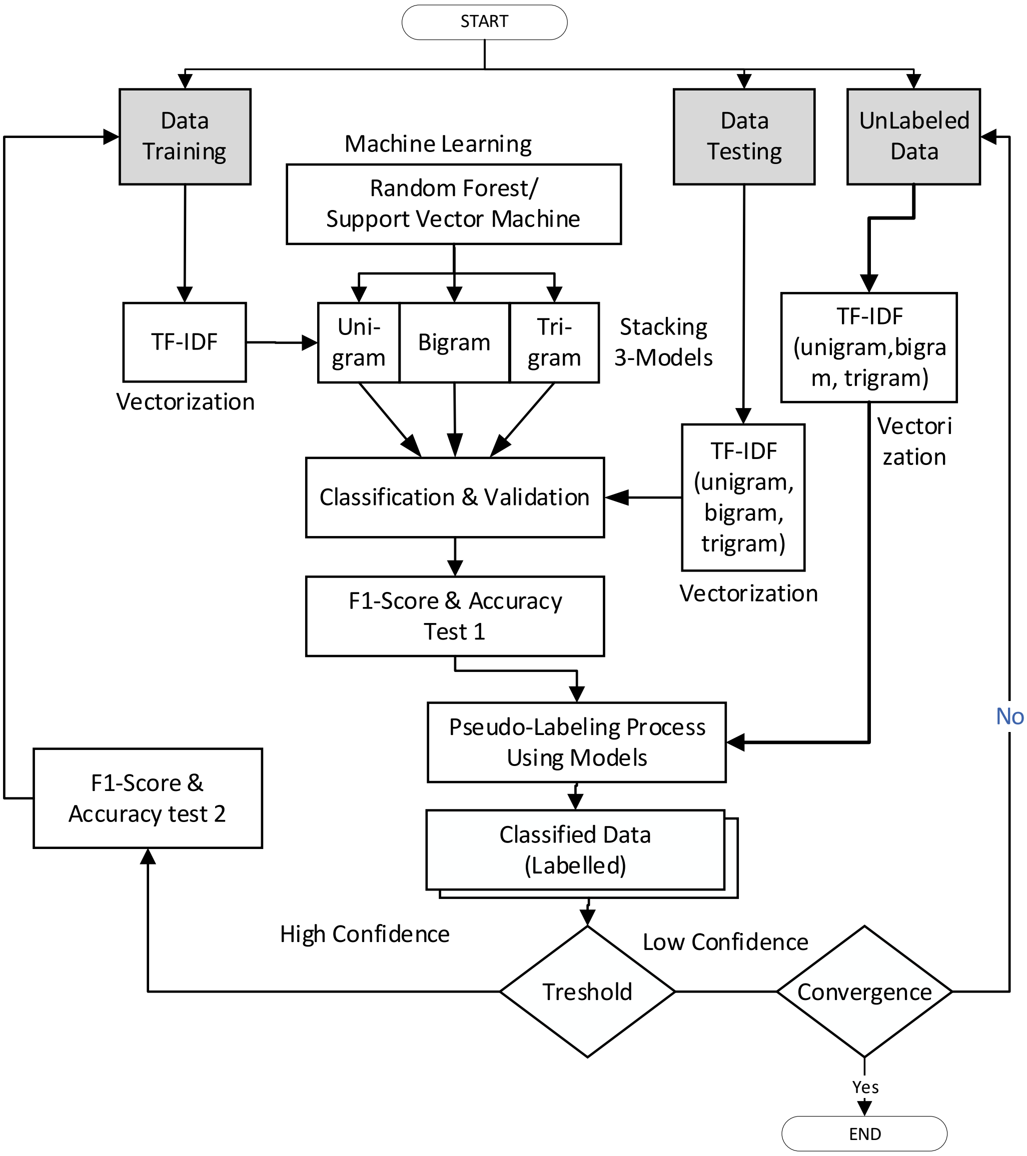

START
Data Training
Machine Learning
Random Forest/ Support Vector Machine
Data Testing
UnLabeled Data
TF-IDF
Uni-gram
Bigram
Tri-gram
Stacking 3-Models
TF-IDF (unigram,bigram, trigram)
Vectorization
Vectorization
Classification & Validation
TF-IDF (unigram, bigram, trigram)
F1-Score & Accuracy Test 1
Vectorization
Pseudo-Labeling Process Using Models
No
F1-Score & Accuracy test 2
Classified Data (Labelled)
High Confidence
Low Confidence
Treshold
Convergence
Yes
END

Fig 3. New SSL-Model Architecture

2.5. Pseudocode

Fig 4 describes the pseudocode of the proposed model. The pseudocode began with the declaration of a threshold number. The next step was to input training data (DT), testing data (DTest), and unlabeled data (UN) on lines 2-4. The DataTraining, Data testing, and Unlabeled dataset would be converted to unigram, bigram, and trigram using TF-IDF methods (lines 6-9). Then, three classifier models would be formed using three training sets and machine learning (RF or SVM) on line 10. In the next step, every classifier validated the data using data testing. Accuracy and F1-score were used as metrics to measure the performance of each model (lines 12-14).

The labeling process was on lines 15-17. The selection process for each new annotated data, whether it was suitable for training data, was on lines 18-24. The process began by checking whether the new annotated data tended to be positive, negative, or neutral based on pseudo-label weights (lines 28-28). If more than the threshold, then, it deserved to be a training data. Otherwise, it would be checked in the next iteration with the new model (formed with the new training data).

```
Threshold=[60%]
READ DT        //Data Train(X,y)
READ DTest   //Data Test(X,y)
READ UN       //Unlabeled Data(X)
ML=['SVM','RF']     //Machine learning
VTestUnigram, VTestBigram, VTestTrigram =TFIDF (DTest, ngram=1,2,3)
Loop Until Convergence OR LEN(UN)==0:
    VTrainUnigram, VTrainBigram, VTrainTrigram = TFIDF(DT, ngram=1,2,3)
    VUnlabeledUnigram, VUnlabeledBigram, VUnlabeledTrigram =TFIDF(UN, ngram=1,2,3)
    Model1, Model2,Model3 = ML.Train(VTrainUnigram, VTrainBigram, VTrainTrigram)
    Result[1], Result[2], Result[3]=Model1.Predict(VTestUnigram, VTestBigram, VTestTrigram )
    Perform[1]=F1Score(Result[1],DTest.y) AND Accuracy(Result[1],DTest.y)
    Perform[2]=F1Score(Result[2],DTest.y) AND Accuracy(Result[2],DTest.y)
    Perform[3]=F1Score(Result[3],DTest.y) AND Accuracy(Result[3],DTest.y)
    Label[1]=Model1.Predict(VUnlabeled_Unigram)
    Label[2]=Model2.Predict(VUnlabeled_Bigram)
    Label[3]=Model3.Predict(VUnlabeled_Trigram)
    For J = 1 to LEN(UN):
        WeightPos=0; WeightNeu=0; WeightNeg=0; Total=0
        For Model=1,3:
                Predicted= Label[Model].RecordNo[J]
                If Predicted=="Positive" Then WeightPos+=Perform[Model]
                If Predicted=="Neutral" Then WeightNeu+=Perform[Model]
                If Predicted=="Negative" Then WeightNeg+=Perform[Model]
                Total+= Perform[Model]
        If WeightPos/Total >= Threshold: Append(UN[J])as "Positive" to DT and Remove(UN[J]) from UN
        If WeightNeu/Total >= Threshold: Append(UN[J])as "Neutral" to DT and Remove(UN[J]) from UN
        If WeightNeg/Total >= Threshold: Append(UN[J])as "Negative" to DT and Remove(UN[J]) from UN
Validate(DT)    //Classify the dataset DT using six models and validate using accuracy and F1Score
```

Fig 4. Pseudocode of Proposed Semi-Supervised Model

2.6. Validation

Confusion Matrix is a performance measurement for machine learning classification. Confusion Matrix output can be two or more classes as in the research [31] and [32]. The confusion matrix compares the actual conditions and predicted results (Table 2).

Table 2: Confusion Matrix for Two Class

		Actual	
		Positive	**Negative**
Predicted	**Positive**	True Positive / TP	False Positive / FP
	Negative	False Negative / FN	True Negative / TN

This research applied two parameters to validate the model: Accuracy and F1 score. Accuracy as the formula in (1) is a ratio of correctly predicted observations i.e. the number of true positive (TP) and true-negative (TN) to the total observations. Total observation is the number of true positives (TP), true-negative (TN), false-positive(FP), and false-negative (FN).

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

$$F1\ Score = 2*(Recall * Precision) / (Recall + Precision) \quad (2)$$

$$Precision = TP/TP+FP \quad (3)$$

$$Recall = TP/TP+FN \quad (4)$$

F1 Score as the formula in (2) is the weighted average of Precision and Recall. F1 Score is usually more useful than accuracy, especially if the result has an uneven class distribution. Precision (3) is the ratio of correctly predicted positive observations (TP) to the total predicted positive observations (TP + FP). High precision relates to the low false positive rate. Recall (Sensitivity) is the ratio of correctly predicted true observations (TP) to all observations in actual class true (TP +FN). Recall is presented in (4).

3. Results and Discussion

3.1. Experiment on US Airline Dataset

The US Airlines dataset was randomly divided into test data and training data. The number of labeled test data for each E1, E2, E3, and E4 was 1464. For data training, four datasets coded as E1, E2, E3, and E4 were prepared. The number of labeled training data (annotated dataset) in E1, E2, E3, and E4 were 2928, 1464, 732, and 366 respectively. The leftover training data was used as the unlabeled data set (unannotated dataset). The baseline model in every experiment E1, E2, E3, and E4 was trained with labeled training data and tested using the labeled test data. Table 3 shows the first experiment, the results of the SSL were processed step by step from the E1 with a 0,6 (60%) threshold using SVM, and Table 4 shows the first experiment using Random Forest.

Table 3. SSL Iteration and Performance in The First Experiment Using SVM

Iteration	Number Of Documents			Accuracy	F1-Score	Information
	Data Training	Data Testing	Unlabelled Data			
Baseline	2928	1410	9758	0,67	0,69	Start Iteration
1	10569	1410	2117	0,73	0,73	
2	12419	1410	267	0,69	0,70	
3	12515	1410	171	0,69	0,70	
4	12527	1410	159	0,69	0,70	
5	12528	1410	158	0,69	0,70	
6	12529	1410	157	0,69	0,70	
7	12529	1410	157	0,69	0,70	Converge, iteration ends

Table 3 explains that the first step (baseline row) is to measure baseline performance. The model built used 2928 training data and classified 1410 test data. The test results showed that the baseline accuracy was 0.67 and the F1-Score was 0.69. In this step, 9758 unlabeled data had not been processed. In the next step, the first iteration, 10569 new training data, which were the sum of the previous training data and annotation results (from the unlabeled dataset), were generated. The new training data was used to create a new classification model. The new model was tested for its performance using data testing, and the results showed an increase in accuracy to 0.73 and the F1 score increased to 0,73. At this step, there were only 2117 unlabeled data sets remaining. In iterations 2 to 7, the explanation is the same as in the first iteration. In the second iteration and so on, the accuracy decreased to 0.69 and the F1-Score to 0.70. The seventh iteration is the last step, the number of unlabeled datasets was 0 document. Accuracy and F1-Score were 0.69 and 0.70, also known as final performance of SSL. The final condition was convergent, i.e. the amount of unlabelled data was the same as the unlabelled data in the previous iteration. The remaining 157 unlabelled data required manual annotation. So far, it could be concluded that SVM was able to increase the accuracy from the baseline (from 0.67 to 0.69) and increase the F1-score from 0.69 to 0.70. The experiment was continued in the Random Forest in Table 3.

Fig 5 is a comparison graph of the performance of each iteration from Table 3. The graph shows the performance increases in the first step of SSL, then decreases and stabilizes in the second step and so on.

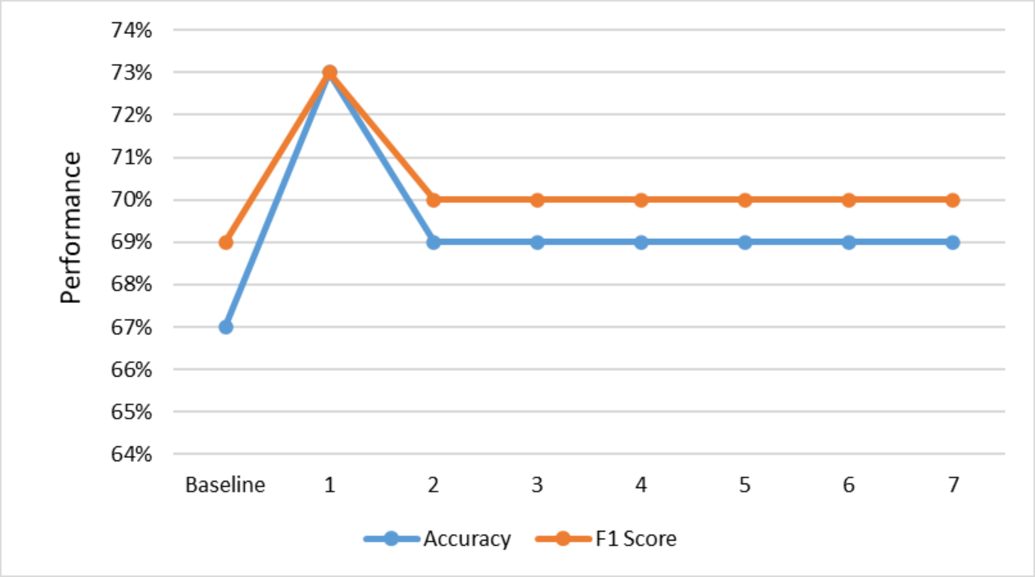


Fig 5. Comparison Graph of Accuracy and F1 Score for Each Iteration Table 3

Fig 5 explains that at the baseline stage to iteration 1, there is an increase in performance Accuracy and F1-Score. This is because the increase in accuracy and F1-score is due to the classifier model formed using labeling from experts. In the second to seventh iteration, the accuracy decreased to 0.69 and the F1-Score to 0.7 because the classifier model was formed using a combined labeling of expert and machine (pseudo-label).

Table 4. SSL Iteration and Performance in The First Experiment Using Random Forest

Iteration	Number Of Documents			Accuracy	F1-Score	Information
	Data Training	Data Testing	Unlabelled Data			
Baseline	2928	1410	9758	0,73	0,73	Start Iteration
1	9464	1410	3222	0,70	0,71	
2	12686	1410	0	0,71	0,72	Iteration ends

Fig 6 shows the performance comparison of each iteration from Table 4. The graph shows that performance decreased in the first step of SSL, then increased in the second step.

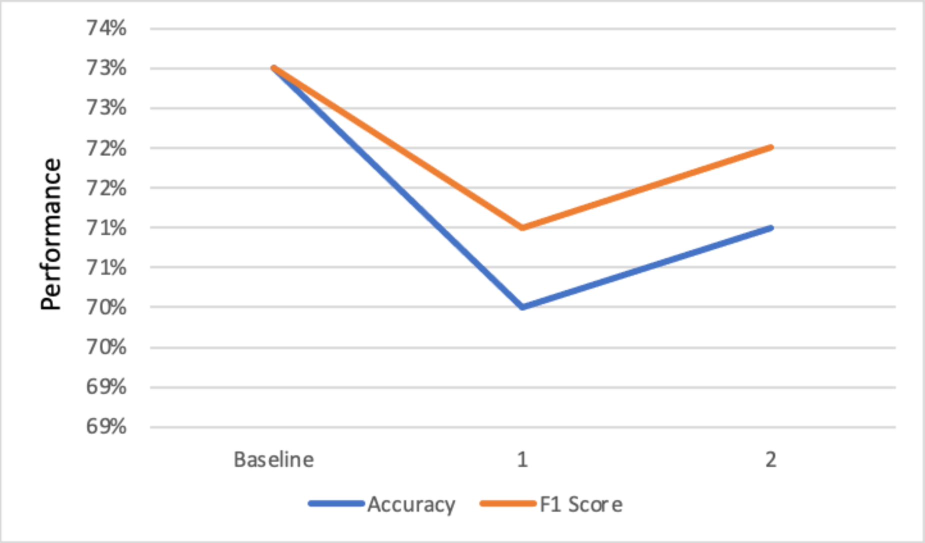


Fig 6. Comparison Graph of Accuracy and F1 Score for Each Iteration Table 4

As in Table 3, Table 4 explains that the first step is to measure baseline performance. The model built used 2928 training data and classify 1410 test data so that the baseline accuracy was 0.73 and the F1-Score was 0.73, higher than the SVM trial. In this step, 9758 unlabeled data had not been processed. In the next step, the first iteration, 9464 new training data were generated and used to create a new classification model. The new model was tested for its performance using data testing, and the results showed a decrease in accuracy to 0.70 and the F1 score decreased to 0.71. Figure 6 explains that from the baseline to iteration 1 there is a decrease in Accuracy and F1-Score performance because the way the random forest model classifier worked was not as good as SVM (on the US Airlines dataset). In the second iteration, the accuracy increased to 0.71 and F1-Score to 0.72 because the RF classifier model was smarter after being formed using a combination of expert and machine labeling (pseudo-label). In this step 3222 unlabeled data sets remained. The second iteration was the last step, the number of unlabeled datasets was 0 document. Accuracy and F1-Score were 0.71 and 0.72, also known as final performance of SSL. The final condition was obtained after all unlabeled data had been successfully annotated. SVM iteration was more selective in the classification process, so that it had more iterations than RF and on US Airlines, and SVM performance was higher than RF.

At baseline, RF was higher than SVM, but there was a decrease in baseline accuracy (from 0.73 to 0.71) and a decrease in F1-score (from 0.73 to 0.72). The advantage was that RF had fewer iterations and all unlabeled data were successfully annotated.

The experiment was continued in scenarios E2, E3, and E4. Accuracy results and F1-scores from all experiments are presented in Table 5. The experiments were still on two machine learning models at the 60% threshold.

Table 5. Accuracy and F1-Score of SSL Models on US Airline Dataset

Experiment (and the number of data training)	Accuracy						F1-Score					
	SVM			Random Forest			SVM			Random Forest		
	Baseline	SSL	Diff	Baseline	SSL	Diff	Baseline	SSL	Diff	Baseline	SSL	Diff
E1 (2928)	0,67	0,69	0,02	0,73	0,71	-0,02*	0,69	0,70	0,01	0,73	0,72	-0,01*
E2 (1464)	0,68	0,69	0,01	0,70	0,70	0	0,69	0,69	0	0,70	0,71	0,01
E3 (732)	0,63	0,65	0,02	0,72	0,65	-0,07*	0,66	0,68	0,02	0,70	0,67	-0,03*
E4 (366)	0,64	0,69	0,05	0,71	0,65	-0,06*	0,66	0,66	0	0,67	0,67	0
Average			0,03			-0,04*			0,005			-0,01*

* Diff polarity negative (-) means there is a decrease in performance

Table 5 explains that the accuracy and F1-score at the baseline of the RF models are higher than in SVM models. The results of semi-supervised learning classification show that the accuracy and F1-score also tend to be linear with the number of training data instances. The difference between the average accuracy of the baseline and the average accuracy of the SSL model in SVM is 0,03 which is better than the RF SSL model (-0.04). The difference between the average F1 score of the baseline and the F1 score of the SSL model in SVM is 0,005 which is better than the RF SSL model (-0.01). SSL models created using SVM tended to provide better accuracy over the baseline. This means that SVM was better at maintaining the performance of the SSL process than RF, but in some experiments RF was higher in performance than SVM.

3.2. Experiment on IMDB Dataset

Similar to the previous experiment, four experimental datasets coded as E1, E2, E3, and E4 were prepared. IMDB dataset was randomly divided into training data and test data in a 9:1 ratio. The number of labeled test data for each E1, E2, E3, and E4 was 5000 (10% of all IMDB data). The number of labeled training data (annotated dataset) in E1, E2, E3, and E4 were 5000, 2500, 1250, and 625 respectively. The leftover training data was used as the unlabeled dataset (as an unannotated dataset). The same as the previous experiment, the baseline model in E1, E2, E3, and E4 was trained with labeled training data without pseudo-label. The baseline model was tested using the labeled test data.

Table 6 shows the first experiment which the results of the SSL were processed step by step from the E1 dataset experiment with a 60% threshold using SVM and Table 7 shows the one used Random Forest. The first step in Table 6 (in baseline line), the model built used 5000 training data and classified 5000 test data so that the baseline accuracy was 0.85 and the F1-Score was 0.85. In this step, 40000 unlabeled data had not been processed at any case. In the next step, in the first iteration, 45000 new training data were generated and used to create new classifier. The new classifier was tested using data testing, and showed an decrease in accuracy to 0.83 and the F1 score decrease to 0,83. The second iteration is the last step, the number of unlabeled datasets was 0 document, known as final performance of SSL

.

Table 6. SSL Iteration and Performance in The First Experiment Using SVM

Iteration	Number Of Documents			Accuracy	F1-Score	Information
	Data Training	Data Testing	Unlabelled Data			
Baseline	5000	5000	40000	0,85	0,85	Start Iteration
1	45000	5000	0	0,83	0,83	Iteration ends

Table 7. SSL Iteration and Performance in The First Experiment Using Random Forest

Iteration	Number Of Documents			Accuracy	F1-Score	Information
	Data Training	Data Testing	Unlabelled Data			
Baseline	5000	5000	40000	0,84	0,84	Start Iteration
1	45000	5000	0	0,80	0,80	Iteration ends

Table 7 explains that in Random Forest the baseline accuracy is 0.84 and the F1-Score is 0.84, lower than SVM model. In the next step, first iteration, 45000 new training data were generated and used to create a new classification model. The new model was tested and the accuracy decrease to 0.80 and the F1 score decreased to 0,80. The second iteration was the last step, the number of unlabeled datasets was 0 document, also known as final performance of SSL. Both methods processed the same number of iterations. The experiment was continued in scenarios E2, E3, and E4 on two machine learning models at the 60% threshold. Accuracy and F1-scores from all experiments are presented in Table 8.

Table 8. Accuracy and F1-Score of SSL Models on IMDB Dataset

Experiment (and the number of data training)	Accuracy						F1-Score					
	SVM			Random Forest			SVM			Random Forest		
	Baseline	SSL	Diff	Baseline	SSL	Diff	Baseline	SSL	Diff	Baseline	SSL	Diff
E1 (2928)	0,85	0,83	-0,02*	0,84	0,80	-0,04*	0,85	0,83	-0,02*	0,84	0,80	-0,04*
E2 (1464)	0,83	0,81	-0,02*	0,82	0,79	-0,03*	0,83	0,82	-0,01*	0,82	0,79	-0,03*
E3 (732)	0,81	0,80	-0,01*	0,81	0,78	-0,03*	0,81	0,81	0	0,80	0,78	-0,02*
E4 (366)	0,80	0,77	-0,03*	0,79	0,75	-0,04*	0,79	0,77	-0,02*	0,78	0,75	-0,03*
Average			-0,02*			-0,03*			-0,01*			-0,03*

* Diff polarity negative (-) means there is a decrease in performance

Table 8 describes SSL-model operations using the IMDB dataset, and presents different results from US Airlines. The accuracy and F1-score at baseline of the SVM models were higher than Random Forest models. The accuracy and F1-score also tended to be linear with the number of training data instances. The difference between the average accuracy of the baseline and the average accuracy of the SSL model in SVM is -0,02 which was better than the RF SSL model (-0.03). The difference between the average F1 score of the baseline and the F1 score of the SSL model in SVM was -0,01 which was better than the RF SSL model (-0.03). In both types of machine learning, there was a decrease in the accuracy of the SSL model to the baseline. However, the main conclusion is that SVM is better at maintaining the accuracy of the SSL process than RF.

This study outperformed Balakrishnan et al's F1 score (on RF gave F1 score = 73.8% and SVM 72.2%) [14]. It also outperformed the F1-Score from research [15] which F1-Score results were 79,039 on B-SVM (SVM model without SSL) and 79.95 on SSSVM (SVM model with bootstrap) when using 1000 labeled data. In this study, on the IMDB dataset, the F1-score results reached 83% for SVM and 80% for RF.

4. Conclusion

This study presents semi-supervised learning for sentiment classification with an ensemble multi-classifier approach to construct an annotated sentiment corpus from US Airlines and IMDB dataset. TF-IDF techniques were implemented to build a vector for modeling the classifier. The results of this study provide several conclusions. The first conclusion is that in SSL the accuracy of the classification is highly dependent on the suitability of the dataset with the machine learning algorithm used. In the IMDB dataset and US Airlines dataset, SVM is better at improving model performance against the baseline. In Airlines dataset, RF is better at achieving baseline performance but fails to maintain model performance. The next research is a sentiment analysis test using several machine learning, datasets, and vectorizers, such as FastText or Word2Vec.

Acknowledgement

The authors would like to thank the Computational Intelligence and Technologies laboratory (CIT Lab) research group, the Center of Advanced Computing Technology (C-ACT), Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka (UTeM), Informatics Engineering Department, Universitas Pembangunan Nasional “Veteran” Yogyakarta Indonesia, and PJP/2020/FTMK/PP/S01791 for their incredible support for this research.

Declaration

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. This research is funded by Universitas Pembangunan Nasional “Veteran” Yogyakarta.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

[1] P. P. Patil, S. Phansalkar, and V. V. Kryssanov, *Topic modelling for aspect-level sentiment analysis*, vol. 828. Springer Singapore, 2019.

[2] 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.

[3] Z. Zhang, J. Han, J. Deng, X. Xu, F. Ringeval, and B. Schuller, “Leveraging Unlabeled Data for Emotion Recognition with Enhanced Collaborative Semi-Supervised Learning,” *IEEE Access*, vol. 6, pp. 22196–22209, 2018, doi: 10.1109/ACCESS.2018.2821192.

[4] A. Alessa and M. F. B, *Analysis Features and TF-IDF Weighting*, vol. 2, no. Cdc. Springer International Publishing, 2018.

[5] V. Kumar and B. Subba, “A Tfidfvectorizer and SVM based sentiment analysis framework for text data corpus,” *26th National Conference on Communications, NCC 2020*, pp. 1–6, 2020, doi: 10.1109/NCC48643.2020.9056085.

[6] A. U. Rehman, A. K. Malik, B. Raza, and W. Ali, “A Hybrid CNN-LSTM Model for Improving Accuracy of Movie Reviews Sentiment Analysis,” *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 26597–26613, 2019, doi: 10.1007/s11042-019-07788-7.

[7] M. HUSSEİN and F. ÖZYURT, “A New Technique for Sentiment Analysis System Based on Deep Learning Using Chi-Square Feature Selection Methods,” *Balkan Journal of Electrical and Computer Engineering*, vol. 9, no. 4, pp. 320–326, 2021, doi: 10.17694/bajece.887339.

[8] 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.

[9] 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.

[10] R. Alahmary and H. Al-Dossari, “A semiautomatic annotation approach for sentiment analysis,” *Journal of Information Science*, 2021, doi: 10.1177/01655515211006594.

[11] A. Alqarafi, A. Adeel, A. Hawalah, K. Swingler, and A. Hussain, *A Semi-supervised Corpus Annotation for Saudi Sentiment Analysis Using Twitter*, vol. 10989 LNAI. Springer International Publishing, 2018.

[12] O. Al-Harbi, “Classifying sentiment of dialectal arabic reviews: A semi-supervised approach,” *International Arab Journal of Information Technology*, vol. 16, no. 6, pp. 995–1002, 2019.

[13] V. D. H. de Carvalho, T. C. C. Nepomuceno, and A. P. C. S. Costa, *An Automated Corpus Annotation Experiment in Brazilian Portuguese for Sentiment Analysis in Public Security*, vol. 384 LNBIP. Springer International Publishing, 2020.

[14] V. Balakrishnan, P. Y. Lok, and H. Abdul Rahim, “A semi-supervised approach in detecting sentiment and emotion based on digital payment reviews,” *Journal of Supercomputing*, vol. 77, no. 4, pp. 3795–3810, 2021, doi: 10.1007/s11227-020-03412-w.

[15] Y. Han, Y. Liu, and Z. Jin, “Sentiment analysis via semi-supervised learning: a model based on dynamic threshold and multi-classifiers,” *Neural Computing and Applications*, vol. 32, no. 9, pp. 5117–5129, 2020, doi: 10.1007/s00521-018-3958-3.

[16] A. Naresh and P. Venkata Krishna, “An efficient approach for sentiment analysis using machine learning algorithm,” *Evolutionary Intelligence*, vol. 14, no. 2, pp. 725–731, 2021, doi: 10.1007/s12065-020-00429-1.

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

[18] A. Rane and A. Kumar, “Sentiment Classification System of Twitter Data for US Airline Service

Analysis,” in *Proceedings - International Conference on Computer Software and Applications*, 2018, pp. 769–773, doi: 10.1109/COMPSAC.2018.00114.

[19] R. S. Kumar, A. F. Saviour Devaraj, M. Rajeswari, E. G. Julie, Y. H. Robinson, and V. Shanmuganathan, “Exploration of sentiment analysis and legitimate artistry for opinion mining,” *Multimedia Tools and Applications*, 2021, doi: 10.1007/s11042-020-10480-w.

[20] Y. Pan, Z. Chen, Y. Suzuki, F. Fukumoto, and H. Nishizaki, “Sentiment analysis using semi-supervised learning with few labeled data,” *Proceedings - 2020 International Conference on Cyberworlds, CW 2020*, pp. 231–234, 2020, doi: 10.1109/CW49994.2020.00044.

[21] M. A. Azim and M. H. Bhuiyan, “Text to emotion extraction using supervised machine learning techniques,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 16, no. 3, pp. 1394–1401, 2018, doi: 10.12928/TELKOMNIKA.v16i3.8387.

[22] S. Mitra and M. Jenamani, “SentiCon: A Concept Based Feature Set for Sentiment Analysis,” in *2018 13th International Conference on Industrial and Information Systems, ICIIS 2018 - Proceedings*, 2018, no. 978, pp. 246–250, doi: 10.1109/ICIINFS.2018.8721408.

[23] P. H. Prastyo, I. Ardiyanto, and R. Hidayat, “Indonesian Sentiment Analysis: An Experimental Study of Four Kernel Functions on SVM Algorithm with TF-IDF,” *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy, ICDABI 2020*, 2020, doi: 10.1109/ICDABI51230.2020.9325685.

[24] S. S. M. M. Rahman, K. B. M. B. Biplob, M. H. Rahman, K. Sarker, and T. Islam, *An investigation and evaluation of N-gram, TF-IDF and ensemble methods in sentiment classification*, vol. 325 LNICST, no. August. Springer International Publishing, 2020.

[25] R. Hendrawan, Adiwijaya, and S. Al Faraby, “Multilabel Classification of Hate Speech and Abusive Words on Indonesian Twitter Social Media,” *2020 International Conference on Data Science and Its Applications, ICoDSA 2020*, 2020, doi: 10.1109/ICoDSA50139.2020.9212962.

[26] M. Aufar, R. Andreswari, and D. Pramesti, “Sentiment Analysis on Youtube Social Media Using Decision Tree and Random Forest Algorithm: A Case Study,” *2020 International Conference on Data Science and Its Applications, ICoDSA 2020*, 2020, doi: 10.1109/ICoDSA50139.2020.9213078.

[27] M. A. Fauzi, “Random forest approach fo sentiment analysis in Indonesian language,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 12, no. 1, pp. 46–50, 2018, doi: 10.11591/ijeecs.v12.i1.pp46-50.

[28] A. R. Alaei, S. Becken, and B. Stantic, “Sentiment Analysis in Tourism: Capitalizing on Big Data,” *Journal of Travel Research*, vol. 58, no. 2, pp. 175–191, 2019, doi: 10.1177/0047287517747753.

[29] M. A. Fauzi, “Word2Vec model for sentiment analysis of product reviews in Indonesian language,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 1, pp. 525–530, 2019, doi: 10.11591/ijece.v9i1.pp525-530.

[30] Y. Li, Y. Lv, S. Wang, J. Liang, J. Li, and X. Li, “Cooperative hybrid semi-supervised learning for text sentiment classification,” *Symmetry*, vol. 11, no. 2, pp. 1–17, 2019, doi: 10.3390/sym11020133.

[31] J. Liang, R. Li, and Q. Jin, “Semi-supervised Multi-modal Emotion Recognition with Cross-Modal Distribution Matching,” *MM 2020 - Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2852–2861, 2020, doi: 10.1145/3394171.3413579.

[32] S. Zhang *et al.*, “Combining cross-modal knowledge transfer and semi-supervised learning for speech emotion recognition,” *Knowledge-Based Systems*, vol. 229, p. 107340, 2021, doi: 10.1016/j.knosys.2021.107340.