https://doi.org/10.48048/tis.2021.680

Natural Disaster on Twitter: Role of Feature Extraction Method of Word2Vec and Lexicon Based for Determining Direct Eyewitness

Mohammad Reza Faisal*, Radityo Adi Nugroho, Rahmat Ramadhani, Friska Abadi, Rudy Herteno and Triando Hamonangan Saragih

Department of Computer Sciences, Lambung Mangkurat University, Banjarbaru 70714, Indonesia

(*Corresponding author's e-mail: reza.faisal@ulm.ac.id)

Received: 23 July 2020, Revised: 19 May 2021, Accepted: 23 May 2021

Abstract

Researchers have collected Twitter data to study a wide range of topics, one of which is a natural disaster. A social network sensor was developed in existing research to filter natural disaster information from direct eyewitnesses, none eyewitnesses, and non-natural disaster information. It can be used as a tool for early warning or monitoring when natural disasters occur. The main component of the social network sensor is the text tweet classification. Similar to text classification research in general, the challenge is the feature extraction method to convert Twitter text into structured data. The strategy commonly used is vector space representation. However, it has the potential to produce high dimension data. This research focuses on the feature extraction method to resolve high dimension data issues. We propose a hybrid approach of word2vec-based and lexicon-based feature extraction to produce new features. The Experiment result shows that the proposed method has fewer features and improves classification performance with an average AUC value of 0.84, and the number of features is 150. The value is obtained by using only the word2vec-based method. In the end, this research shows that lexicon-based did not influence the improvement in the performance of social network sensor predictions in natural disasters.

Keywords: Feature extraction, Natural disaster, Text classification, Word2vec, Lexicon

Introduction

A sensor is a device or subsystem that functions to detect events or changes in an environment and send that information to other devices or subsystems. In the field of disaster management, sensors have an essential role. Sensors are placed at specific locations to read the situation. Then the data will be sent periodically to a data storage center for processing. Data processing aims to determine which data states disaster occurred, such as an earthquake, forest fire, or flood. Furthermore, the data center can send an early warning to those who need it. The results of data processing can also be used to monitor the situation before and during a disaster.

The term social network sensor has been used in many studies [1,2]. The social network sensor has the same function as the sensor described earlier. Social network censors use online users to send states where the user is located when disaster strikes. They send this message or information to social media like Twitter. Then the classification process is carried out to determine the category of the tweet. In the field of disaster management, messages are categorized into 3 namely (i) natural disaster information sent by direct eyewitnesses, (ii) natural disaster information posted by non-eyewitnesses, and (iii) information that is not related to natural disaster [3]. After knowing which disaster eyewitnesses sent messages, the temporal and spatial information is extracted from the tweet to determine the time and location of the natural disaster.

The process of classifying Twitter messages into the 3 categories mentioned above is called sentiment analysis. Until now, there has been a lot of research on sentiment analysis. Sentiment analysis consists of the following steps: (i) feature extraction, i.e., converting Twitter messages into structured data, (ii) creating a classification model, structured data is processed by the classification algorithm to acquire the model, (iii) using the model to predict categories from new Twitter messages. The overall performance of the social network sensor is determined by the chosen technique or method in the 1st and 2nd steps.

Feature extraction is a step that must be done in sentiment analysis because the classification algorithm can only process structured data. And the classification performance can be influenced by the feature extraction technique [4]. The feature extraction technique commonly applied in research related to disaster management is vector space representation in matrix term-frequency [3,5]. This technique is easy to use, but the number of features cannot be determined because it depends on the number of unique words in text data. It can produce high dimension data that will make a long computational time when processed by a classification algorithm and requires high hardware specifications. Based on this problem, a study of feature extraction techniques was carried out to convert text into structured data with low dimensions. The resulting structured data is tested to find out which methods can improve the performance of the classifier.

Generally, there are 3 types of feature extraction methods on text, i.e., vector space representation, word-embedding vectors based, and lexicon-based.

Imran *et al.* [5] conducted a multiclass classification of Twitter data related to natural disasters from 2013 to 2015. The feature extraction technique used is the vector space representation type, namely unigram and bigram. The classification algorithms used are Naïve Bayes, Random Forest, and Support Vector Machine (SVM). For the evaluation of trained models, 10-fold cross-validation is used with the performance value of AUC = 0.5.

Tarmizi *et al.* [6] present the task of Author Identification for KadazanDusun language by using tweets as the source of data. The feature extraction used is a combination of n-grams which n is from 1 to 5. Then processed each data with Naïve Bayes and SVM. This study shows combination unigram and 3-gram with SVM as the classification algorithm gave the best result with an accuracy of 80.17 %.

Zahra *et al.* [3] conducted a classification to determine data on Twitter messages from eyewitnesses related to forest fires, floods, hurricanes, and earthquakes. The feature extraction technique used is unigram and bigram and then combined with features of domain-expert analysis results. In addition, feature selection is also performed using information gain. While the classification algorithm used is Random Forest. The author also balances the amount of data per class with the SMOTE method to improve the performance of the classification model. The performance model is evaluated using 10-fold cross-validation with an average AUC = 0.9.

The number of features generated by vector space representation techniques is high dimensional data, as shown in **Table 1**.

Table 1 Structured data generated by vector space representation techniques.

No	Method	Dataset	Number of features
1	Unigram	Floods	6013
2	Unigram	Earthquakes	2456
3	Unigram	Hurricanes	4072
4	Unigram	Forest Fires	3905
5	Bigram	Floods	19987
6	Bigram	Earthquakes	7948
7	Bigram	Hurricanes	14390
8	Bigram	Forest Fires	12985

The 2nd type of feature extraction method is word embedding vectors based. In research that was conducted by Orkphol and Yang [7] used 3 ways to create sentence vectors, namely (i) the summation of all Word2Vec vectors of each word in the sentence, (ii) the average of all word vectors, and (iii) weighting all word vectors using inverse document frequency (IDF).

The above technique is also used for the similarity of a document [8]. This technique is also used to extract Twitter data features for crisis event classification cases [9]. Word-embedding vector models were generated by the Word2Vec, GloVe, and FastText algorithms. Nine feature extraction techniques were carried out to produce structured data with the number of features, as seen in **Table 2**. The classification algorithm used is Random Forest, KNN, SVM, and GNB with classified classification performance, which is between 0.7 - 0.92.

Table 2 shows the number of features resulted from techniques that use word-embedding vectors that can be defined. In addition, the data dimension is lower compare to the output of the methods in **Table 1**.

Table 2 Structure data generated by word embedding vectors-based feature extraction.

No	**Method**	**Dataset**	**Number of features**
1	Word2Vec	Crisis Tweet	300
2	CrisisW2V	Crisis Tweet	300
3	GloVe	Crisis Tweet	25, 50, 100, 200
4	CrisisGloVe	Crisis Tweet	100
5	FastText	Crisis Tweet	300
6	CrisisFastText	Crisis Tweet	300
7	SIF	Crisis Tweet	100
8	InferSent	Crisis Tweet	4096
9	tfSent	Crisis Tweet	512

These techniques may generate low dimension data. But the technique of making sentence vectors such as summation and average of all word-embedding vectors can produce the same sentence vector values. If this technique is used as a social network sensor to detect natural disaster events, it can decrease the sensor's performance.

The 3rd type of feature extraction on text is lexicon-based. This technique classifies the sentiment of a term based on the dictionary provided. There are many lexicon dictionaries developed by researchers such as Socal Google [10], NRC emotion [11], Hu and Liu [12], Slangd [13], Indonesia Lexicon[14], and SentiWord [15]. But the words in those dictionaries are limited, some words may not be found in a particular dictionary, and it can make incomplete information.

This research aims to look for which feature extraction techniques can generate low dimension data. The 2nd and 3rd type of feature extraction is used to fulfill this purpose. But both methods may have a weakness, so we also try to combine features from each technique to find out if it affects improving the performance of the classification algorithm.

There are 2 motivations for this study based on the explanation above. Social network sensors that have high accuracy are critical to be developed. So that humans can use it as an early warning system and monitor conditions when a disaster occurs. However, the social network sensors currently being developed still use vector space representation-based feature extraction, which produces high-dimensional data. This research aims to create new feature extraction by combining vector-based and lexicon-based word-embedding methods to have data with far fewer features. This study aims to determine whether data with fewer features can produce the same or better social network sensor performance.

Materials and methods

Materials

In this study, we use datasets from the research of K. Zahra *et al.* [3] that are shown in **Table 3**. There are 4 datasets from Twitter data related to natural disasters such as forest fire, earthquake, hurricane, and flood. Every dataset has 3 class labels. The number of samples in every class is not the same or imbalanced. The class labels or category are:

- direct eyewitness, this class label is messages from the eyewitness.
- none eyewitness, messages of this class label are from non-eyewitness.
- don't know, messages which are not related to the natural disaster.

Table 3 Raw Dataset.

No	Dataset	Category		
		Direct eyewitness	**Don't know**	**None eyewitness**
1	Forest Fires	189	432	1,379
2	Earthquakes	1,600	200	200
3	Hurricanes	465	336	1,199
4	Floods	627	822	551

Tools and methods

R package

We use the R programming language as a tool. The word-embedding vector-based extraction feature uses the wordVectors package [16], and the lexicon-based feature extraction uses the sentiment package [17]. Package stats contain hierarchical clustering functions which are used for clustering [18]. The randomForest package is used to create a classification model [19]. Calculation of the performance of classification models using the pROC package [20].

Random forest

The random forest algorithm was proposed by Breiman in 2001 [21]. It is a supervised learning algorithm. As a general-purpose classification and regression system, it has been highly effective. The method, which combines multiple randomized decision trees and averages their predictions, has been shown to perform well in situations where the number of variables exceeds the number of observations.

The random forest algorithm is based on the decision tree algorithm. The Random Forest algorithm has 2 stages: The first is to generate a random forest. The second is to make a prediction using the random forest classifier generated in the 1st stage. The pseudocode of the entire procedure is listed below:

1. Randomly select "K" features from total "m" features where k << m.
2. Among the "K" features, calculate the node "d" using the best split point.
3. Split the node into daughter nodes using the best split.
4. Repeat the a to c steps until "l" number of nodes has been reached.
5. Build forest by repeating steps a to d for "n" number times to create "n" number of trees.

The next step is predicting by using the random forest classifier. The pseudocode for random forest prediction is shown below.

1. Takes the test features and predicts the outcome using the rules of each randomly generated decision tree, then saves the predicted result (target).
2. Calculate the number of votes for each predicted target.
3. Consider the high-voted predicted target as the final prediction from the random forest algorithm.

Word2Vec

Word2vec was created, patented, and published in 2013 by a team of researchers led by Tomas Mikolov at Google [22]. Word2vec is a natural language processing technique. The word2vec algorithm learns word associations from an enormous corpus of text using a neural network model. Once learned, a model like this can detect synonyms and recommend additional terms for a sentence. As the name suggests, word2vec associates each particular word with a specific set of numbers known as a vector.

Word2vec can make highly accurate guesses about a word's meaning based on previous appearances if given enough data, usage, and contexts. These guesses may be used to determine a word's connection with other words (for example, "man" is to "boy" what "woman" is to "girl"), or to cluster and classify documents by topic.

Ward's hierarchical agglomerative clustering

Clustering is a strategy for grouping similar data points together such that the points in the same group are more alike than the points in other groups. The set of similar data points is called a cluster. One of the most common and straightforward clustering techniques is hierarchical clustering. There are 2 kinds of clustering techniques: agglomerative and divisive.

Agglomerative hierarchical clustering technique, initially, each data point is considered as an individual cluster. Similar clusters merge with other clusters in each iteration until 1 cluster or K cluster is created. The pseudocode for this clustering technique is shown below.

```
1. Compute the proximity matrix.
2. Let each data point be a cluster.
3. Repeat: Merge the 2 closest clusters and update the proximity matrix.
4. Until only a single cluster remains.
```

Ward's is the only one among the agglomerative clustering methods based on a classical sum-of-squares criterion, producing groups that minimize within-group dispersion at each binary fusion [23]. Since Ward's 1st description in a 1963 publication, the Ward error sum of squares hierarchical clustering approach has been commonly used.

Research procedure

Figure 1 is our research procedure. It shows how the classifier work in social network sensors to identify the natural disaster tweet from Twitter. There are 4 steps in this procedure: (i) pre-processing, (ii) feature extraction, (iii) learning by using training data to create a model, (iv) use the model to do testing and validation. The output of this process is the performance report.

The 1st step is pre-processing. In this step, we clean the raw datasets by using some processes such as eliminating punctuation, numbers, non-alphanumeric characters, multiple spaces, and non-Latin characters.

The 2nd step is the most crucial in our study. We applied feature extraction techniques on cleaned Twitter data to generate structured data. Two types of feature extraction are used: (i) word-embedding vector-based and (i) lexicon-based. These techniques are used to made new features.

The word-embedding vector-based feature extraction aims to generate sentence vectors using the word vectors. We obtain the word vector by using word2vec. Word2vec function is used to create the corpus to a fixed-size vector for each word [24]. In this research, we use all the datasets in **Table 3** as a corpus. Each vector produced consists of 100 features.

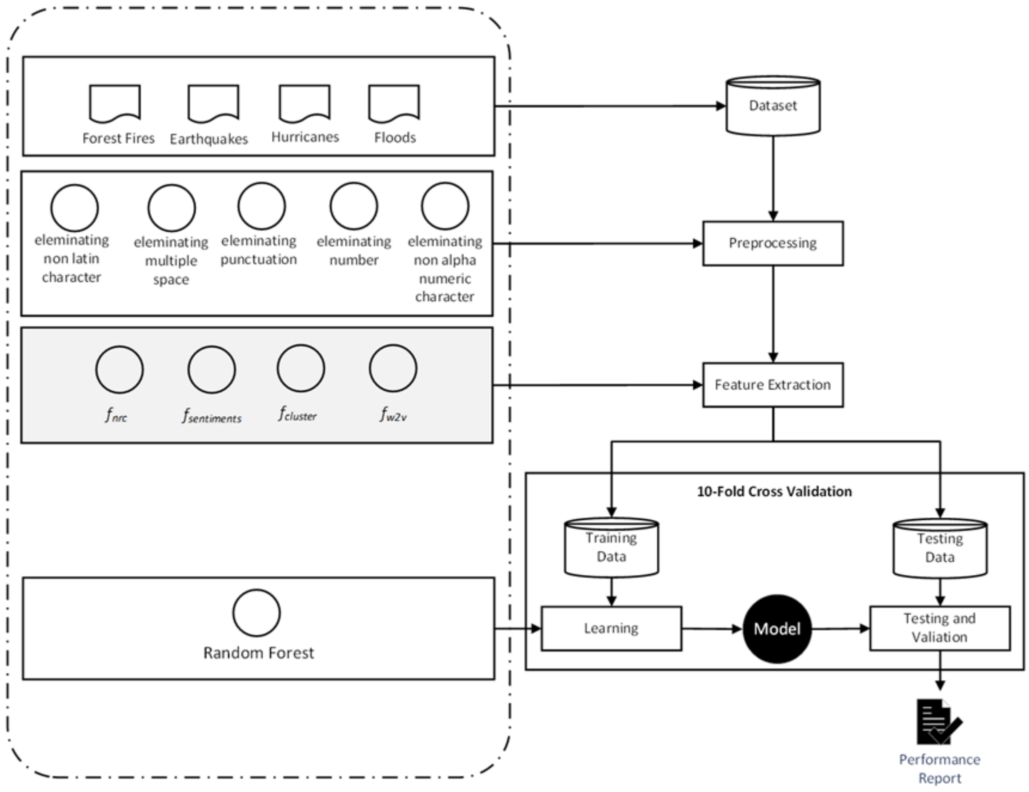


Figure 1 Research procedure.

The 1st word-embedding vector-based feature extraction is the technique to generated sentence vectors by using generated word vectors. In this research, we create a sentence vector with the average of all vectors in the sentence. The formula of this technique can be seen in Eq. 1 [7], where S is the sentence from the Twitter message. n_s is the number of words in the sentence S. $v(\cdot)$ is a function to get the vector of the word S_i. The result of the function $f(S)$, it has a vector with 100 features.

$$f_{w2v}(S) = \frac{1}{n_s}\sum_{i=0}^{n_s} v(S_i) \tag{1}$$

The 2nd word-embedding vector-based feature extraction is the bag of centroid. The previous method has a weakness because there is a possibility that the number of vectors from the word vector or the value of the sentence vector has the same or similar value as other sentences. Therefore we try to collect similar words. Then the vectors of similar terms are added. We name this technique the bag of centroid. The 1st step of this technique is clustering all word vectors from word2vec word vectors. The clustering algorithm used is Ward that is one of the hierarchical clustering algorithms [25]. Since we do not yet know the best number of word groups, we try several values of word groups. The number of clusters made are 5, 10, 15, 20, 25, 50, 75, 100, 125, 150 and 175. Next, generate structured data by calculating the frequency of occurrence of words for each cluster. The formula of this technique is shown in Eq. 2. Where $n_{cluster}$ is the number of clusters. $freq(\cdot)$ is a function to check whether the word S_j belongs to cluster i, if it is true then the number of frequencies of cluster i will be increased by 1. The structured data that is generated by this technique has $n_{cluster}$ features.

$$f_{cluster}(S) = \bigcup_{i=1}^{n_{cluster}} \left(freq_i\left(S_{j=1}^{n_s}\right)\right) \tag{2}$$

The other type of feature extraction is lexicon-based feature extraction. This technique is commonly used in sentiment analysis research to determine positive and negative comments. However, there has been no research using this technique on social network sensors for natural disasters. This technique calculates the sentiment value of a sentence by checking the sentiment value of each word in the lexicon dictionary. Then calculate the sum of all these sentiment values. In this study, we use several lexicon dictionaries, namely: Socal Google [10], NRC emotion [11], Hu and Liu [12], Short Jockers, Jockers, Loughran McDonald, SenticNet, Slangd [13], and SentiWord [15]. This feature extraction technique is described in Eq. 3. dic is the lexicon dictionary. $senti_{dic}(\cdot)$ is a function to find the sentiment value of the word S_i in the lexicon dictionary dic. The structured data generated by this function has 9 features.

$$f_{sentiment}(S) = \bigcup_{dic}\left(\sum_{i=1}^{n_s} senti_{dic}(S_i)\right) \tag{3}$$

Also, we calculate the frequency of word types from the NRC lexicon dictionary. Type words consist of anger, anticipation, disgust, fear, joy, sadness, surprise, trust, negative, and positive. Eq. 4 is a function of this feature extraction technique. $type$ is the word type. $freq_{type}(\cdot)$ is a function to calculate the frequency of those word types in S sentences. This function produces structured data with ten features.

$$f_{nrc}(S) = \bigcup_{type}\left(freq_{type}\left(S_{i=1}^{n_s}\right)\right) \tag{4}$$

We made several combinations of output from those techniques to generated new features used to build classification models. Combining several results from several feature extraction techniques has been common in many existing research. The goal is to get new and more complete features. From the results of these studies, it is known that the combination of these features can improve classification performance. Those features are shown in **Table 4**.

Table 4 Features generated by merging feature extraction techniques.

No	Name	Feature extraction techniques	# Features
1	F1	f_{w2v}	100
2	F2	$f_{w2v} \cup f_{sentiment} \cup f_{nrc}$	119
3	F3	$f_{w2v} \cup f_{cluster}$	105 - 275
4	F4	$f_{w2v} \cup f_{sentiment} \cup f_{nrc} \cup f_{cluster}$	124 - 294

After structured data are generated, data are divided into training data and testing data. The ratio of each data is based on 10-fold cross-validation. Then the 3rd step is applied; it uses training data in the learning process to make the classification model using Random Forest. And the last step is testing and validation. This step uses the classification model to predict the class label of the testing data. The predicted class label and actual class label are calculated to obtain the value of AUC (Area Under the Curve) ROC (Receiver Operating Characteristic) [26] for each class. AUC ROC or AUC value is commonly used to evaluate classification model performance. We analyze each classification model performance to find out which feature extraction technique helps the classification model get the best performance.

Results and discussion

There are 3 objectives in our experiments. The 1st objective is to know the performance of each classification model that is built with described techniques in the previous section. The 2nd objective is to see the influence of additional features from the bag of centroids and lexicon-based. And the last, we check the comparison of the number of features and performance value of classification between our proposed features and features generated using vector space representation techniques (unigram and bigram).

Structured data

Structured data is generated using the feature extraction techniques that are mentioned in **Table 4**. We created 96 structured data that are shown in **Table 5**.

Table 5 Generated Structured Data Using Features F1, F2, F3 and F4.

Structured data	Raw dataset	Feature name	# Cluster	# Features
D1	Forest Fires	F1	-	100
D2		F2	-	119
D3 - D13		F3	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	105 - 275
D14 - D24		F4	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	124 - 294
D25	Earthquakes	F1	-	100
D26		F2	-	119
D27 - D37		F3	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	105 - 275
D38 - D48		F4	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	124 - 294
D49	Hurricanes	F1	-	100
D50		F2	-	119
D51 - D61		F3	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	105 - 275
D62 - D72		F4	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	124 - 294

Structured data	Raw dataset	Feature name	# Cluster	# Features
D73	Floods	F1	-	100
D74		F2	-	119
D75 - D85		F3	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	105 - 275
D86 - D96		F4	5, 10, 15, 20, 25, 50, 75, 100, 125, 150, 175	124 - 294

Performance of classification models

Next, we make 96 classification models with Random Forest and evaluated them with 10-fold cross-validation. The prediction results of each model that uses the same feature extraction are combined, and then the performance is calculated. It means that the prediction results of the classification model that are made by using the D1, D25, D49 and D73 structured data will be combined, and then the classification performance is calculated. This result represents the classification model performance if we use the F1 feature extraction technique.

Another example is combining the prediction results of the classification model created using the D13, D37, D61 and D85 datasets. This result shows the classification model performance if we use the F3 feature extraction technique using 175 clusters. **Table 6** shows the results of the methods we used.

Table 6 Performance of classification (AUC).

No	Feature name	#Features	Category		
			Direct eyewitness	Don't know	None eyewitness
1	F1	100	0.846	0.796	0.865
2	F2	119	0.844	0.796	0.865
3	F3 5	105	0.844	0.801	0.864
4	F3 10	110	0.843	0.801	0.864
5	F3 15	115	0.845	0.802	0.866
6	F3 20	120	0.845	0.801	0.868
7	F3 25	125	0.844	0.798	0.866
8	F3 50	150	0.846	0.802	0.868
9	F3 75	175	0.846	0.799	0.868
10	F3 100	200	0.847	0.799	0.865
11	F3 125	225	0.845	0.797	0.866
12	F3 150	250	0.846	0.799	0.867
13	F3 175	275	0.847	0.797	0.868
14	F4 5	124	0.844	0.800	0.866
15	F4 10	129	0.844	0.800	0.866
16	F4 15	134	0.844	0.800	0.865
17	F4 20	139	0.844	0.798	0.866
18	F4 25	144	0.845	0.799	0.866
19	F4 50	169	0.845	0.797	0.866
20	F4 75	194	0.845	0.797	0.867
21	F4 100	219	0.845	0.798	0.866
22	F4 125	244	0.845	0.798	0.868

No	**Feature name**	**#Features**	**Category**		
			Direct eyewitness	**Don't know**	**None eyewitness**
23	F4 150	269	0.844	0.795	0.866
24	F4 175	294	0.847	0.801	0.868

Based on the results above, it shows the results of the F1 and F2 techniques. And F3 and F4 techniques have several classification performance results based on several variations in the number of clusters. **Figure 2** shows the classification performance comparison (AUC). In this picture, it is not clear which cluster provides the best performance. And **Figure 3** shows the average AUC of the 3 classes. It shows a technique that provides the best classification performance value, namely the F3 approach with 50 clusters.

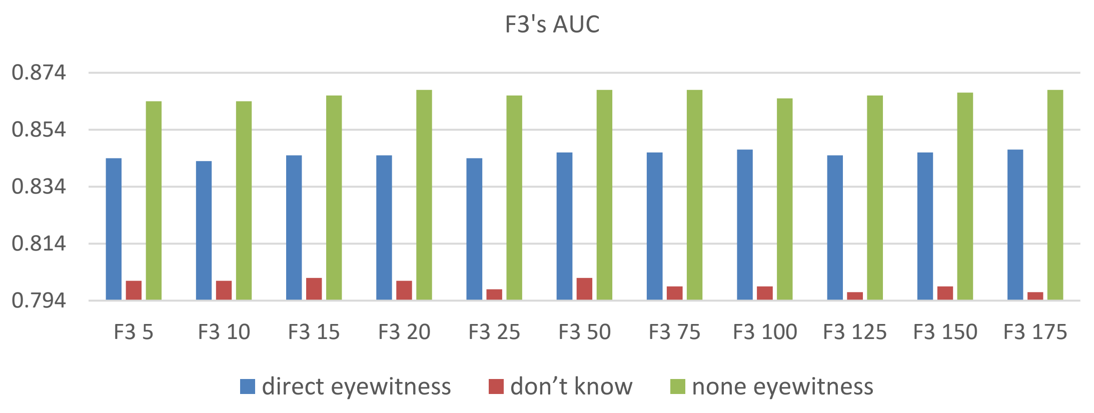
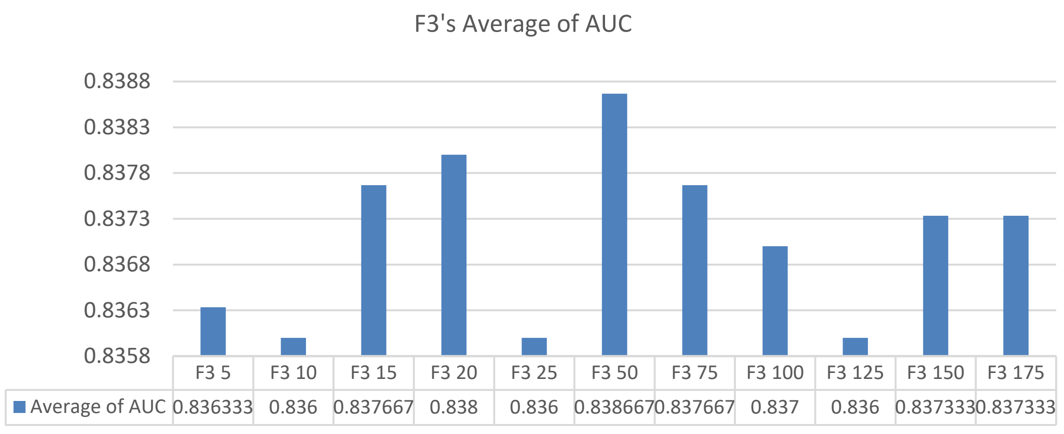


Figure 2 Classification performance of F3.



	F3 5	F3 10	F3 15	F3 20	F3 25	F3 50	F3 75	F3 100	F3 125	F3 150	F3 175
🟦 Average of AUC	0.836333	0.836	0.837667	0.838	0.836	0.838667	0.837667	0.837	0.836	0.837333	0.837333

Figure 3 Average classification performance of each class by using the F3 technique.

Figure 4 shows the classification performance comparison (AUC). But **Figure 5** shows which approach provides the best classification performance. It shows the average AUC of the 3 classes and shows that the F4 technique provides the best classification performance value, namely the F4 approach with 125 and 175 clusters. However, we chose cluster 175 as a better value because the AUC for the direct eyewitness class is higher.

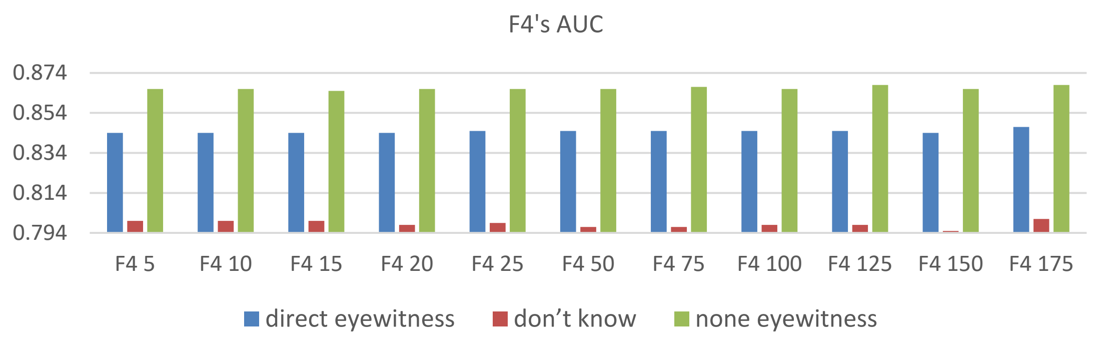


Figure 4 Classification performance of F4.

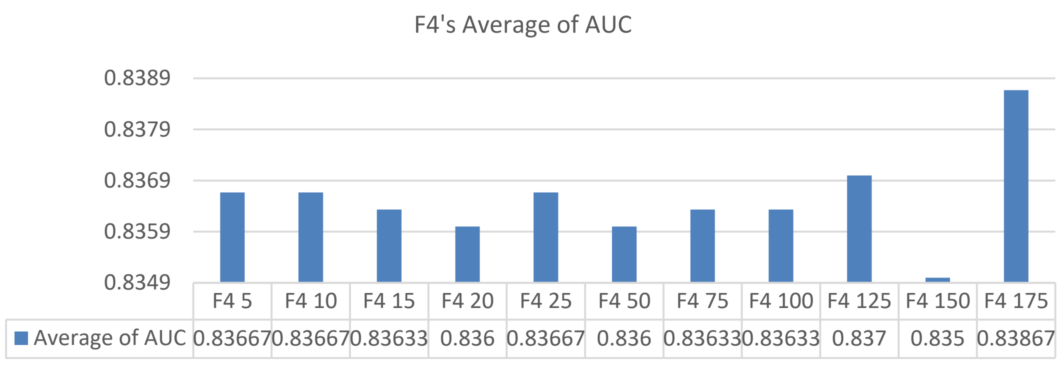


Figure 5 Average classification performance of each class by using the F4 technique.

Performance comparison with other techniques

We compare our results to results from the unigram and bigram feature extraction techniques. The comparison result can be seen in **Table 7**. The results of our methods are listed in rows 2 - 4. The 1st row shows the performance of the average of all vectors in the sentence (F1). And the 2nd row is the performance value by merging features from the average of all vectors in the sentence and lexicon-based (F2). The 3rd row shows the best result of combining features from the average of all vectors in the sentence and bag of centroid (F3). The best work of F3 is acquired if the number of clusters is 50. The 4th row is the best result of merging the feature average of all vectors in the sentence, bag of centroid, and lexicon-based (F4). The best result of F4 is acquired from using 175 clusters. Row 5 and 6 are the classification performance using structured data resulting from the existing feature extraction method. **Figure 6** indicates the comparison chart of the AUC per class compared to the approach that we use with current feature extractions. **Figure 7** shows the comparison of the mean AUC.

From **Figure 7**, we can see that additional features from lexicon-based could not significantly influence the performance of the average of all vectors in the sentence. The result can be seen from comparing performance between F1 and F2. Another proof could also be seen from comparing the performance of F3 and F4. On the other hand, additional features from the bag of centroid have an impact on the performance of the classification technique of the average of all vectors in the sentence. The proof can be seen from the comparison average AUC values of F1 and F3 and F2 and F4 in **Figure 7**.

Table 7 Comparison of our result to result of n-gram techniques.

No	Feature Name	#Features	Performance (AUC)		
			Category		
			Direct eyewitness	Don't know	None eyewitness
1	F1	100	0.846	0.796	0.865
2	F2	119	0.844	0.796	0.865
3	F3 50	150	0.846	0.802	0.868
4	F4 175	294	0.847	0.801	0.868
5	Unigram	6013	0.832	0.795	0.841
6	Bigram	19987	0.853	0.798	0.812

We used the feature extraction technique to give better value in predicting don't know category and none eyewitness category with the highest AUC value of 0.802 and 0.868. While the n-gram technique only has an AUC value of 0.798 and 0.812 for both mentioned categories. On the other hand, to predict the eyewitness category, our approach shows the highest AUC value of 0.847, which is lower than the AUC value compared to the n-gram technique. If we compare the average AUC in **Figure 7**, all the methods we propose have a better performance than the existing methods.

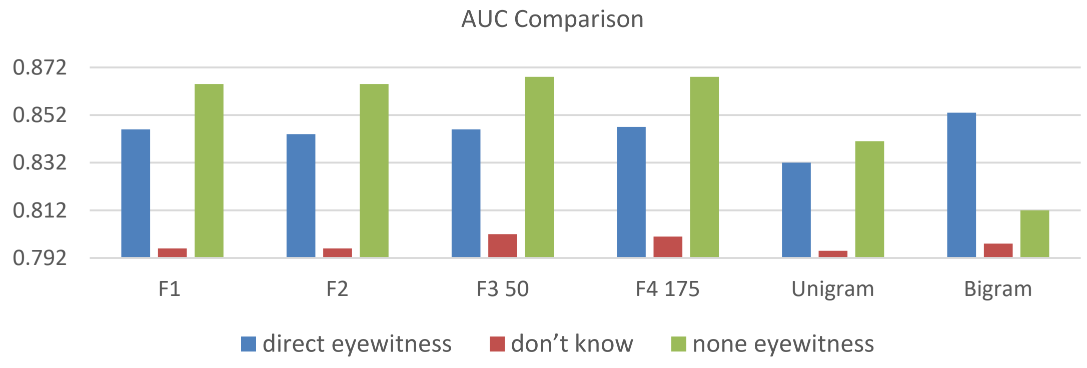
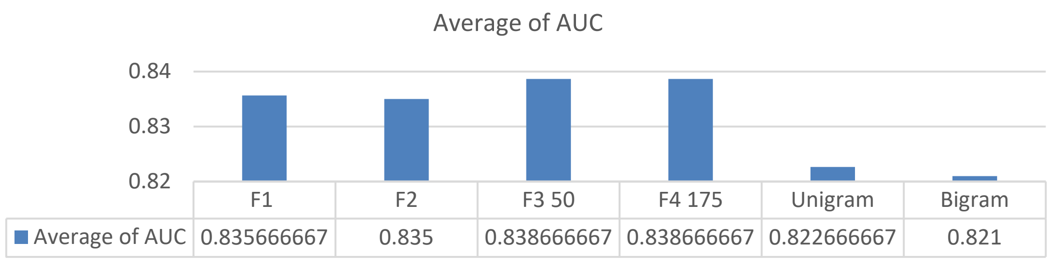


Figure 6 AUC Comparison with existing feature extraction method.



	F1	F2	F3 50	F4 175	Unigram	Bigram
Average of AUC	0.835666667	0.835	0.838666667	0.838666667	0.822666667	0.821

Figure 7 Comparison average of AUC.

Column #Features in **Table 7** show the number of features of each technique. A chart that compares the number of features can be seen in **Figure 8**. It shows that our methods generated structured data with lower dimensions if it corresponds to n-gram techniques. Our approach only generates data with between 119 to 294 features. And the n-gram technique, which could produce maximum features 6013 and 19.987.

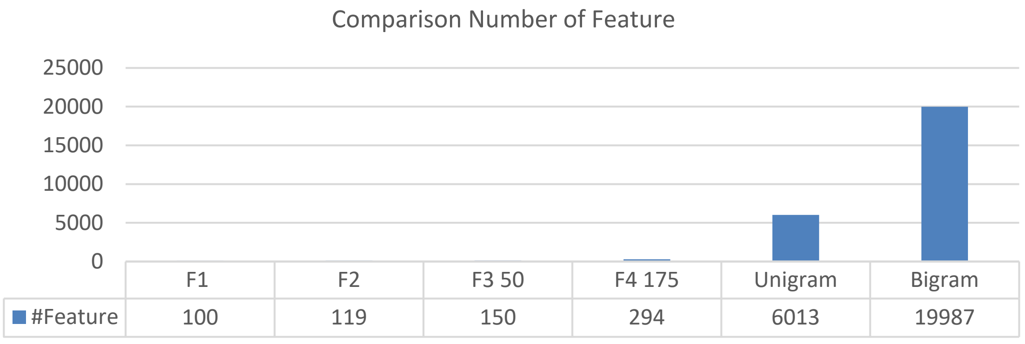


Figure 8 Comparison number of features.

Conclusions and future work

Our research aims to find a feature extraction technique that can replace the vector space representation technique. We saw from this approach is produces high dimension data and generates an arbitrary number of features. The number of features depends on the number of words in the corpus. As proof, the unigram technique yields a feature count between 2,456 to 6,013 and the AUC is 0.823. The bigram technique produces a higher number of features, namely between 7,948 to 19,987 and the AUC is 0.821. We also found that the training process with this high dimension data requires a long computation time of about 48 h.

We explored the word embedding vectors-based technique in this study because it produces low features. The word-embedding method that we use in this study is Word2Vec. The generated features are fixed length, only100 features. In this research, we tried to complete sentence vectors built with Word2Vec with features developed by the lexicon-based (F2) feature extraction function. This function adds 19 features. However, these features did not improve the classification performance but instead decreased it by 0.08 %. Our study confirmed that the existing Lexicon dictionaries are not suitable for social network sensors in cases of natural disasters. The dictionaries can be used to classify negative and positive sentences only. So it is necessary to make a new dictionary for natural disasters to classify the categories direct eyewitness, none eyewitness, and don't know.

Another feature extraction technique that we use is the bag of centroid. We add the features produced by this technique to the F1 and F2 techniques to produce the F3 and F4 techniques. The bag of centroid technique succeeded in getting the best classification performance compared to the F1, F2, Unigram, and Bigram techniques. Classification performance improvement is about 0.36 % better when compared to F1 and F2 techniques. And 2.11 % better when compared to the Unigram and Bigram techniques.

The best technique we propose produces a very much less number of features when compared to the feature extraction base on vector space representation. The number of features decreased by about 96 % when compared to the unigram technique. And a decrease of about 99 % when compared to the Bigram technique. In addition, the time used for the training process is much faster, which is less than 1 h.

This study confirmed that feature extraction based on a word-embedding vector can significantly reduce the number of features and provide better classification performance. However, there is still a lot of room for improvement. In this research, we only use a word embedding method, namely word2vec. Other methods that scientists have developed are Glove and Fasttext. In future research, we will use these 2 methods to determine their effect on improving classification performance. Another possible way to improve classification performance is by changing the classification method. The method we will use in the following research is the Convolutional Neural Network to conduct text classification.

Acknowledgments

The authors would like to thank Computer Science, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University, Indonesia, for the support of this research.

References

[1] Y Kryvasheyeu, H Chen, E Moro, P Van Hentenryck and M Cebrian. Performance of social network sensors during hurricane sandy. *PLoS One* 2015; **10**, 1-19.

[2] NA Christakis and JH Fowler. Social network sensors for early detection of contagious outbreaks. *PLoS One* 2010; **5**, e12948-e12948.

[3] K Zahra, M Imran, and FO Ostermann. Automatic identification of eyewitness messages on twitter during disasters. *Inf. Process. Manag.* 2020; **57**, 102107.

[4] MR Faisal, B Abapihi, NG Nguyen, B Purnama, MK Delimayanti, D Phan, FR Lumbanraja, M Kubo and K Satou. Improving protein sequence classification performance using adjacent and overlapped segments on existing protein descriptors. *J. Biomed. Sci. Eng.* 2018; **11**, 126-43.

[5] M Imran, P Mitra and C Castillo. Twitter as a lifeline: Human-annotated Twitter corpora for NLP of crisis-related messages. *In:* Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC 2016 - Portoroz, Slovenia. 2016, p. 1638-43.

[6] N Tarmizi, S Saee, D Hanani and A Ibrahim. Author identification for under-resourced language Kadazandusun. Indonesian *J. Electr. Eng. Comput. Sci.* 2020; **17**, p. 248-55.

[7] K Orkphol and W Yang. Word Sense Disambiguation Using Cosine Similarity Collaborates with Word2vec and WordNet. *Future Internet* 2019; **11**, 114.

[8] F Nurifan, R Sarno and CS Wahyuni. Developing corpora using word2vec and wikipedia for word sense disambiguation. *Indonesian J. Electr. Eng. Comput. Sci.* 2018; **12**, 1239-46.

[9] H Li and X Li, D Caragea and C Caragea. Comparison of word embeddings and sentence encodings as generalized representations for crisis tweet classification tasks. *In:* Proceedings of the Information Systems for Crisis Response and Management Asia Pacific Conference, Wellington, New Zealand, 2018.

[10] PD Turney and ML Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.* 2003; **21**, 315-46.

[11] SM Mohammad and PD Turney. Crowdsourcing a word-emotion association lexicon. *Comput. Intell.* 2013; **29**, 436-65.

[12] M Hu and B Liu. Mining and summarizing customer reviews. *In:* Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA. 2004, p. 168-77.

[13] L Wu, F Morstatter and H Liu. SlangSD: Building, expanding and using a sentiment dictionary of slang words for short-text sentiment classification. *Lang. Resour. Eval.* 2018; **52**, 839-52.

[14] MA Ayu, SS Wijaya and T Mantoro. An automatic lexicon generation for Indonesian news sentiment analysis: A case on governor elections in Indonesia. *Indonesian J. Electr. Eng. Comput. Sci.* 2019; **16**, 1555-61.

[15] S Baccianella, A Esuli and F Sebastiani. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. *In:* Proceedings of Proceedings of the 7th International Conference on Language Resources and Evaluation, Valletta, Malta. 2010.

[16] B Schmidt and J Li. WordVectors: Tools for creating and analyzing vector-space models of texts. R package version 2.0, Available at: http://github.com/bmschmidt/wordVectors, accessed May 2020.

[17] TW Rinker. 2019, Sentiment: Calculate text polarity sentiment. Buffalo, New York, Available at: http://github.com/trinker/sentimentr, accessed May 2020.

[18] R Core Team. R: A language and environment for statistical computing. Vienna, Austria. Available at: https://www.r-project.org, accessed May 2020.

[19] A Liaw and M Wiener. Classification and regression by random forest. *R. News*. 2002; **2**, 18-22.

[20] X Robin, N Turck, A Hainard, N Tiberti, F Lisacek, JC Sanchez and M Müller. pROC: An open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinform.* 2011; **12**, 77.

[21] L Breiman. Random forests. *Mach. Learn.* 2001; **45**, 5-32.

[22] T Mikolov, K Chen, G Corrado and J Dean. Efficient estimation of word representations in vector space. Available at: https://arxiv.org/abs/1301.3781, accessed April 2020.

[23] F Murtagh and P Legendre. Ward's hierarchical agglomerative clustering method: Which algorithms implement Ward's criterion? *J. Classif.* 2014; **31**, 274-95.

[24] T Mikolov, I Sutskever, K Chen, GS Corrado and J Dean. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* 2013; **2**, 3111-19.

[25] J Ward. Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* 1963; **58**, 236-44.

[26] F Gorunescu. *Data mining: Concepts, models and techniques*. Vol XII. Springer, Berlin, 2011.