JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 27 (2023) 145–153
Received: 25 November 2022
Accepted: 7 December 2022

A comparison of word embedding-based extraction feature techniques and deep learning models of natural disaster messages classification

Porównanie technik wyodrębniania cech opartych na osadzeniu słów oraz modeli głębokiego uczenia w klasyfikacji wiadomości o klęskach żywiołowych

Irwan Budiman, Mohammad Reza Faisal*, Friska Abadi, Dodon Turianto Nugrahadi, Muhammad Haekal

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

Abstract

The research aims to compare the classification performance of natural disaster messages classification from Twitter. The research experiment covers the analysis of three-word embedding-based extraction feature techniques and five different models of deep learning. The word embedding techniques that are used in this experiment are Word2Vec, fastText, and Glove. The experiment uses five deep learning models, namely three models of different dimensions of Convolutional Neural Network (1D CNN, 2D CNN, 3D CNN), Long Short-Term Memory Network (LSTM), and Bidirectional Encoder Representations for Transformer (BERT). The models are tested on four natural disaster messages datasets: earthquakes, floods, forest fires, and hurricanes. Those models are tested for classification performance.

Keywords: Twitter; Natural disaster; CNN; LSTM; BERT

Streszczenie

Badanie ma na celu porównanie skuteczności klasyfikacji wiadomości o klęskach żywiołowych z Twittera. Eksperyment badawczy obejmuje analizę technik ekstrakcji cech opartych na osadzeniu trzech słów oraz pięciu różnych modeli głębokiego uczenia. Techniki osadzania słów używane w tym eksperymencie to Word2Vec, fastText i Glove. Eksperyment wykorzystuje pięć modeli głębokiego uczenia, a mianowicie trzy modele o różnych wymiarach konwolucyjnej sieci neuronowej (1D CNN, 2D CNN, 3D CNN), oraz dwie sieci: Long Short-Term Memory Network (LSTM) oraz Bidirectional Encoder Representations for Transformer (BERT). Modele zostały prztestowane na czterech zestawach danych dotyczących klęsk żywiołowych, a mianowicie trzęsień ziemi, powodzi, pożarów lasów i huraganów. Modele te przetestowano pod kątem wydajności klasyfikacji.

Słowa kluczowe: twitter; klęska żywiołowa; CNN; LSTM; BERT

*Corresponding author

Email address: reza.faisal@ulm.ac.id (M. R. Faisal)

1. Introduction

The existence of social media currently plays an essential role in assisting in every activity in the disaster management cycle. In the pre-disaster stage, social media can be used as an early warning before a disaster occurs [1]. At the scene when the natural disaster occurred, eyewitnesses shared information about the situation at that time. It can be used by volunteers or the government to deal with the impact of disasters. Whereas in the post-disaster stage, social media users share messages containing information on relief that has been carried out or information on locations that have not received assistance [2].

Natural disaster messages on social media are categorized into three: *eyewitness* and *non-eyewitness*, and *don't-know* [3]. Messages of the *eyewitness* category are natural disaster messages posted by eyewitnesses at the location when the disaster occurred. Messages in the *non-eyewitness* category are messages about natural disasters uploaded by users who are not eyewitnesses. In contrast, a message in *don't-know* category is a message in which there are words related to natural disasters, but the meaning is not about natural disasters.

Utilization of social media messages related to natural disasters for natural disaster management can be maximized with the help of artificial intelligence. Artificial intelligence can help find natural disaster messages faster [4]. The artificial intelligence system will classify social media messages into the three categories that are mentioned above.

The word embedding-based feature extraction technique is formed by the concatenation of word vectors into 1-dimensional data (1D) [5], [6]. Sentance vectors can be formed by arranging word vectors into a matrix (2D) [7]. Three 2D data created by each word embedding technique such as Wod2vec, Glove and fastText can be combined into 3 layers to form 3-dimensional data (3D) [8]. The output of the feature extraction process is structured data.

The deep learning method that can process multi-dimensional structured data is the Convolutional Neural Network (CNN) [9]–[11]. For text classification with 1D CNN with feature extraction technique based on word2vec [5]. The application of the 2D CNN technique to classify forest fire messages produces a good accuracy of 81.97%[7]. This study used three-word embedding techniques to create 2D data, namely word2vec, fastText and Glove. The application of text data classification with 3D CNN is made by combining 2D data based on word embedding consisting of three layers based on word embedding techniques word2vec [12], Glove [13] and fastText [8].

Another deep learning technique that is commonly used to classify text is the Long Short-Term Memory Network (LSTM). For the case of sentiment analysis on the IMDB dataset [14], the classification performance obtained using the LSTM model works better than the 1D CNN model. In this study, 1D concatenated data were used from the Word2Vec vector and the 1D CNN model. Meanwhile, the LSTM model uses input data from the tokenised form, which is then converted by an embedding layer based on Word2Vec. For the case of classifying natural disaster messages using deep learning models, the performance of the LSTM model is better than CNN [15]. The CNN model used in this study uses 2D data from the Glove vector and the 2D CNN model. And the LSTM model uses input data from the tokenized form, which is then converted by a Glove-based embedding layer.

Bidirectional Encoder Representations for Transformer (BERT) [16] is a recently popular deep learning method. BERT has achieved state-of-the-art results in a broad range of NLP tasks because of its ability to understand words more thoroughly [17]. BERT can provide a richer linguistic structure because linguistic knowledge is stored in hidden states and on attention maps [18].

The explanation above has provided knowledge of the methods used to carry out feature extraction and classification in the case of sentiment analysis and classification of natural disaster messages. However, it is necessary to carry out a comprehensive comparative study of these methods to obtain knowledge of the technique that can provide the best classification performance in the case of natural disaster messages. This research is done to answer following questions:

1. What are the classification performance of 1D CNN, 2D CNN, and 3D CNN models using the three-word embedding techniques in the feature extraction process?
2. What is the classification performance of the LSTM model using the three-word embedding techniques in the feature extraction process?
3. What is the classification performance of the BERT model?

Existing research generally only uses a word embedding technique for feature extraction. This research also combines feature extraction results based on Word2Vec, Glove, and fastText for processing with 1D CNN, 2D CNN, 3D CNN, and LSTM models. The aim is to determine whether combining those techniques can improve the natural disaster message classification performance.

This report is divided into five sections: 1) Introduction, 2) Dataset and method section explaining the dataset and classification algorithms that are used, 3) Research implementation section explaining the steps in the implementation of this study, 4) Result, and the last section is 5) Conclusions.

2. Dataset

The natural disaster message dataset used in this research comes from research [3]. Details about this dataset can be seen in Table 1.

Table 1: The natural disaster dataset

Dataset	Class Label	#Messages
Earthquakes	eyewitness	1600
	dont-know	200
	non-eyewitness	200
Floods	eyewitness	627
	dont-know	822
	non-eyewitness	551
Forest fires	eyewitness	189
	dont-know	432
	non-eyewitness	1379
Hurricanes	eyewitness	465
	dont-know	336
	non-eyewitness	1199

Each dataset has three categories or class labels: *eyewitness*, *dont-know*, and *non-eyewitness*. Examples of natural disaster messages from each class label can be seen in Table 2.

Table 2: Samples of natural disaster messages

No	Message	Class Label
1	We're a family pulled from a flood	eyewitness
2	I'm ready for these earthquake memes	dont-know
3	Houston streets flood again, dampening July 4th celebrations https://t.co/3I1aOZkNBx https://t.co/4i5kHBcRjo	non-eyewitness

The first message is from natural disasters such as floods and earthquakes uploaded by eyewitnesses when the disaster occurred. The secon message is message that contain the words earthquake and flood, but the meaning is not about natural disasters. While the last message is about natural disasters from the news that are re-shared by Twitter users.

3. Research implementation

The steps in the implementation of the research can be seen in Figure 1.

3.1. Text Normalization & Word Padding

Four natural disaster message datasets were normalized with steps commonly performed in text classification cases: removing double spaces, punctuation marks, numbers and non-alphanumeric characters [19]. The four text data that have been normalized are used as input for the BERT method to create a classification model.

Clean text data is then counted as the number of words in each message. After that, each message in each dataset is equated with the word padding based on the mean value.

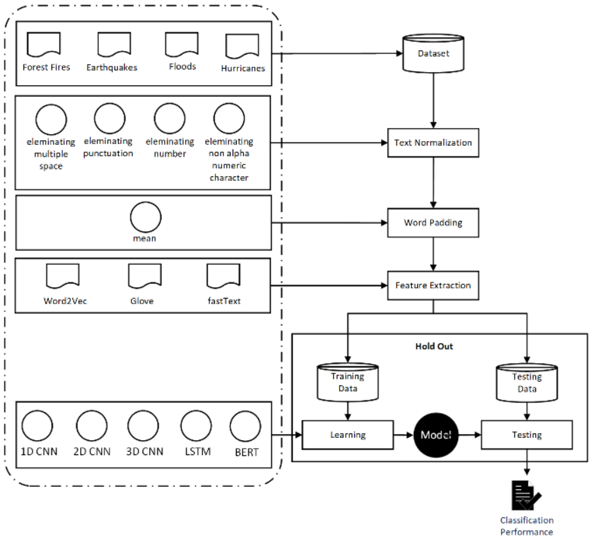

Figure 1: Steps of research implementation.

3.2. Feature Extraction

The next stage is feature extraction. There are three groups of techniques based on the dimensions of the output data, namely, one dimension (1D), two dimensions (2D), and three dimensions (3D). Each of these techniques will use three popular word embedding techniques used in classification studies, namely Word2ve [12], fastText [20], Glove [13], [21]. Data structures with different dimensions are created using the three-word embedding models. The formation of 1D data is explained as follows.

v_i	=	V1	V2	V2	…	V98	V99	V100

If v_i is the vector of a word, and n is the number of words in a sentence, then the 1D data is the sentence vector v_s which is formed by combining all word vectors (Formula 1). The value of n corresponds to the number of words searched for with a statistically based word padding technique, namely the mean.

$$v_s = \bigcup_{i=1}^{n} v_i \qquad (1)$$

Formula 2 shows how to create 1D data by combining the three-word embedding techniques.

$$v_{all} = v_{s\,word2vec} \cup v_{s\,fastText} \cup v_{s\,Glove} \qquad (2)$$

The feature extraction results in this way produce structured data with 1D dimensions, as seen in Table 3. From these results, 16 structured data were obtained, which were used as input to create a classification model using the 1D CNN method.

Table 3: 1D structured data

Input dataset	Word embedding method	Dimension of 1D structured data
Earthquakes	Single word embedding technique	900
	Union of 3 word embedding techniques (All)	2700
Floods	Single word embedding technique	1600
	Union of 3 word embedding techniques (All)	4800
Forest fires	Single word embedding technique	1200
	Union of 3 word embedding techniques (All)	3600
Hurricanes	Single word embedding technique	1300
	Union of 3 word embedding techniques (All)	3900

The formation of 2D data with each word embedding technique is explained as follows. If v_1, v_2, to v_n re word vectors resulting from a number of word embedding techniques m and n is formed by creating a two-dimensional matrix $m \times n$ as shown in Figure 2. Where m is 100.

v_1	=	V1	V2	V2	…	V98	V99	V100
v_2	=	V1	V2	V2	…	V98	V99	V100
…	…	…	…	…	…	…	…	…
v_n	=	V1	V2	V2	…	V98	V99	V100

Figure 2: 2D Data.

This research also proposes the formation of 2D data from a combination of three-word embedding techniques. The combined 2D data can be seen in Figure 3. The 2D data is a two-dimensional matrix $m \times N$, where m is 100 and N is $3 \times n$.

word2vec	V1	V2	V2	…	V98	V99	V100
	V1	V2	V2	…	V98	V99	V100
	…	…	…	…	…	…	…
	…	…	…	…	…	…	…
	V1	V2	V2	…	V98	V99	V100
fastText	V1	V2	V2	…	V98	V99	V100
	V1	V2	V2	…	V98	V99	V100
	…	…	…	…	…	…	…
	…	…	…	…	…	…	…
	V1	V2	V2	…	V98	V99	V100
Glove	V1	V2	V2	…	V98	V99	V100
	V1	V2	V2	…	V98	V99	V100
	…	…	…	…	…	…	…
	…	…	…	…	…	…	…
	V1	V2	V2	…	V98	V99	V100

Figure 3: 2D data combined with three word embedding techniques.

The feature extraction results in this way produce structured data with 2D dimensions, as seen in Table 4. From these results, 16 structured data are obtained, which are used as input to create a classification model using the 2D CNN and LSTM methods.

Table 4: 2D structured data

Input Dataset	Word Embedding Method	Dimension of 2D structured data
Earthquakes	Single word embedding technique	9 x 100
	Union of 3 word embedding techniques (All)	27 x 100
Floods	Single word embedding technique	16 x 100

Input Dataset	Word Embedding Method	Dimension of 2D structured data
	Union of 3 word embedding techniques (All)	48 x 100
Forest fires	Single word embedding technique	12 x 100
	Union of 3 word embedding techniques (All)	36 x 100
Hurricanes	Single word embedding technique	13 x 100
	Union of 3 word embedding techniques (All)	39 x 100

To create 3D data by combining 2D data from three different word embedding techniques. There are two ways of generating 3D data. The way of forming the first 3D data can be seen in Figure 4. 3D data type 1 is generated in 3 layers. The first layer is 2D data from the word2vec technique, followed by 2D data from the fastText and Glove techniques.

If 3D data type 1 is represented as a matrix, the dimensions are$m \times n \times z$. Where m is 100, n is the number of words in a sentence, and z is 3. The method of forming 3D data type 1 follows the method of forming 3D image data consisting of 3 RGB color channels.

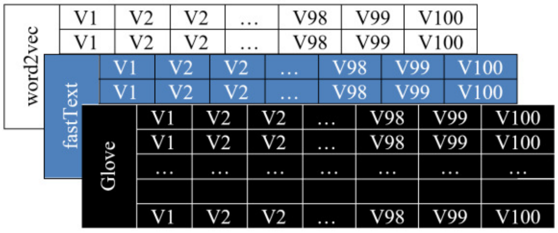

Figure 4: Data 3D type 1.

The way to generate 3D data type 2 is to follow the formation of 3D video. Video are a set of frames or images. For this study, a frame is formed by three vectors of a word from three-word embedding techniques. Then the next layer represents the second word, and so on. So that this data will have n layers. If 3D data type 2 is defined as a matrix, the dimensions are $m \times z \times n$. Where m is 100, z is the number of word embedding techniques, namely 3. n is the number of words in the sentence.

The feature extraction results in this way produce structured data with 3D dimensions, which can be seen in Table 5.

Table 5: 3D structured data

Input Dataset	Word Embedding Method	Dimension of 3D structured data
Earthquakes	Type 1	100 x 9 x 3
	Type 2	100 x 3 x 9
Floods	Type 1	100 x 16 x 3
	Type 2	100 x 3 x 16
Forest fires	Type 1	100 x 12 x 3
	Type 2	100 x 3 x 12
Hurricanes	Type 1	100 x 13 x 3
	Type 2	100 x 3 x 13

3.3. Classification

Three deep learning methods are used in this research: CNN, LSTM, and BERT.

Convolutional Neural Network (CNN) is an artificial neural network used initially in image recognition and processing [22]. The input and output of each stage are in the form of an array called a feature map. The output of each stage is a feature map of the processing results from all input locations. Each stage consists of three layers: the convolutional layer, the activation layer, and the pooling layer [22]. The convolutional layer is the first layer that receives direct input data to the architecture. The convolutional layer performs the convolution operation on the previous layer's output. The purpose of convolution on data is to extract features from the input data. Pooling layer is reducing the size of the matrix by using a pooling operation. Two types of pooling are often used: average pooling and max pooling. The activation function is a node that is added at the end of the output of each neural network. The activation function, also known as the transfer function, is used to determine the neural network output. In the CNN architecture, the activation function lies in the final computation of the feature map output or after the convolution or pooling calculation process to produce a feature pattern. Several kinds of activation functions that are often used in research include the sigmoid, tanh, Rectified Liniear Unit (ReLU), Leaky ReLU (LReLU) dan Parametric ReLU [23].

The result of these three layers is a feature map in the form of a multi-dimensional array. The feature map is then processed by the flatten operation to become a vector. Then the vector is processed by the fully connected layer. The fully connected layer is where all the previous layer's activated neurons are connected to the neurons in the next layer [24]. Each activation of the prior layer needs to be converted into one-dimensional data before it can be linked to all neurons. The Fully-Connected layer is usually used in the Multi Layer Perceptron method to process data so that it can be classified.

Long Short-Term Memory is a variant of Recurrent Neural Network (RNN). LSTM can conduct training and overcome vanishing gradient problems which are difficult for RNNs [25]. LSTM was created with the aim of overcoming the hidden layer problem. LSTM can learn long patterns from sequential data because it prevents vanishing gradient situations. However, LSTM still has the same principle as RNN and what differentiates it from RNN is the cell content. RNN is simple because with cells that only contain a layer of neurons with the tanh activation function. LSTM becomes more complex because the cell's contents are more than one layer of neurons. There is a layer of neurons called gates [26].

Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained contextual word representation model based on MLM (Masked Language Model), using two-way Transformers [18]. The BERT model architecture is a multi-layer bidirectional transformer encoder-decoder structure. Transformers follow this overall architecture using self-attention and point-wise stacked, fully connected encoders and decoders. There are two steps in the performance of the BERT framework, namely pre-training and fine-tuning [18]. BERT pre-training does not use the traditional left-to-right or right-to-left method but uses Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) for pre-training data. MLM fills in the blanks, where the model uses the context word around the mask token to predict what word should be, while NSP is the prediction of the next sentence with the two models given. After pre-training data, BERT will perform fine-tuning where fine-tuning is initialized with previously trained parameters, and all fine-tuning parameters use labeled data from downstream tasks.

Model building implementation uses the Python programming language with the Keras and Tensorflow libraries. In this study, three CNN models were made, namely 1D CNN, 2D CNN and 3D CNN. Table 6 shows the parameters used in each model in this study.

Table 6: Parameters of CNN models

Algorithm	Parameters
1D CNN	Input Shape layer • Output = max_word
	Word embedding layer • embedding word, embedding vector, embedding matrix, max word • Output = max_word, 100
	4 convolutional layers = Conv_1, Conv_2, Conv_3, Conv_4 Convolutional 1D • Kernel size = n; n = {1, 2, 3, 5} • Activation=Tanh • Kernel initializer = orthogonal • Kernel regularizer=L1,L2 • Bias initializer = glorot • Output = max_word, 128 Max pooling 1D • Pool size = max_word-n; n = {1, 2, 3, 5} • Padding = valid • Output = max_word, 128; 1, 128 Dropout • P=0.25 • Output = 1, 128
	Concatenate • Input = Conv_1, Conv_2, Conv_3, Conv_4 • Output = 4, 128
	Flatten_1 • Input = flatten • Output = 512
	Dropout_1 • P = 0.35 • Output = 512
	Dense_1 • Units = 3 • Activation = softmax • Output = 3
2D CNN	Input Shape layer • Output = max_word
	Word embedding layer • embedding word, embedding vector, embedding matrix, max word • Output = max_word, 100
	4 convolutional layers = Conv_1, Conv_2, Conv_3, Conv_4 Conv_1 Convolutional 2D • Kernel size= n, embedding vector; n = {1, 2, 3, 5} • Activation=Tanh • Kernel initializer = orthogonal • Kernel regularizer=L1,L2 • Bias Initializer = glorot • Output = max_word, 1, 128 Max pooling 2D • Pool size = max_word-n, 1; n = {1, 2, 3, 5} • Padding = valid • Strindes = 1, 1 • Output = max_word, 1, 128; 1, 1, 128 Dropout • P=0.25 • Output = 1, 1, 128
	Concatenate • Input = Conv_1, Conv_2, Conv_3, Conv_4 • Output = 4, 1, 128
	Flatten_1 • Input = flatten • Output = 512
	Dropout_1 • P = 0.35 • Output = 512
	Dense_1 • Units = 3 • Activation = softmax • Output = 3
3D CNN Type 1	Input Shape layer • Output = max_word
	4-word embedding layers = Word embedding 1, Word embedding 2, Word embedding 3, Word embedding 4 Word embedding • embedding word, embedding vector, embedding matrix, max word • Output = max_word, 100
	Concatenate • Input = Word_embedding_1, Word_embedding_2, Word_embedding_3 • Output = max_word, 300
	4 convolutional layers = Conv_1, Conv_2, Conv_3, Conv_4 Conv_1 Convolutional 3D • Kernal size = **n, embedding vector, 3; n = {1, 2, 3, 5}** • Activation=Tanh • Kernel initializer = orthogonal • Kernel Regularizer=L1,L2 • Bias Initializer = glorot • Output = max_word, 1, 128 Max pooling 3D • Pool size = max_word-n, 1, 1; n = {1, 2, 3, 5}

Algorithm	Parameters
	• Padding = valid • Strindes = 1, 1 • Output = max_word, 1, 1, 1, 128; 1, 1, 1, 128 Dropout • P=0.25 • Output = 1, 1, 1, 128
	Concatenate • Input = Conv_1, Conv_2, Conv_3, Conv_4 Output = 4, 2, 1, 128
	Flatten_1 • Input = flatten • Output = 512
	Dropout_1 • P = 0.35 • Output = 512
	Dense_1 • Units = 3 • Activation = softmax • Output = 3
3D CNN Type 2	Input Shape layer Output = max_word
	4-word embedding layers = Word embedding 1, Word embedding 2, Word embedding 3, Word embedding 4 Word embedding • embedding word, embedding vector, embedding matrix, max word • Output = max_word, 100
	4 convolutional layers = Conv_1, Conv_2, Conv_3, Conv_4 Conv_1 Convolutional 3D • Input = **n, embedding vector, 1; n = {1, 2, 3, 5}** • Activation=Tanh • Kernel initializer = orthogonal • Kernel Regularizer=L1,L2 • Bias Initializer = glorot • Output = max_word, 1, 128 Max pooling 3D • Pool size = (max_word*3)-n, 1, 1; n = {1, 2, 3, 5} • Padding = valid • Strindes = 1, 1 • Output = max_word, 1, 1, 1, 128; 1, 1, 1, 128 Dropout • P=0.25 • Output = 1, 1, 1, 128
	Concatenate • Input = Conv_1, Conv_2, Conv_3, Conv_4 • Output = 4, 2, 1, 128
	Flatten_1 • Input = flatten • Output = 512
	Dropout_1 • P = 0.35 • Output = 512
	Dense_1 • Units = 3 • Activation = softmax • Output = 3

Meanwhile, the LSTM and BERT models were built using the parameters shown in Table 7.

Table 7: parameters of LSTM and BERT model.

Algorithm	Parameters
LSTM	Input Shape layer • Output = max_word
	Word embedding • embedding word, embedding vector, embedding matrix, max word • output = max_word, 100
	Bidirectional • unit = 256 • output = 256
	Dense 1 • unit = 64 • activation = Relu • Kernel Regularizer = L1, l2 • Bias = Glorot_uniform • Output = 64
	Dense 2 • unit = 32 • activation = Relu • Kernel Regularizer = L1, l2 • Bias = Glorot_uniform • Output = 32
	Dropout 1 • P = 0.35
	Dense 3 • Unit 3 • activation = softmax • Output = 3
BERT	Input_ids • Input = max_word • Output = max_word
	Attention_mask • Input = max_word • Output = max_word
	Tf_Bert_model • Input = input_ids, attention_mask • Output = TFBaseModelOutput
	LSTM • Unit = 32 • Output = 32
	Batch normalization • Output = 32
	Dense 1 • unit = 16 • activation = Relu • Kernel Regularizer = L1, l2 • Bias = Glorot_uniform • Output = 16
	Batch normalization • Output = 16
	Dense 2 • unit = 8 • activation = Relu • Kernel Regularizer = L1, l2 • Bias = Glorot_uniform • Output = 8
	Dropout 1 • P = 0.3
	Dense 3 • Unit 3 • activation = softmax • Output = 3

Table 1 shows the difference in the number of samples from each class so that it is known that this is a case of imbalanced data classification. So that the classification performance used in this research is F1 Score and ROC AUC.

Results

This research consists of 15 experiments for each dataset. So that the total number of experiments carried out is 60 experiments. Figure 5 shows a comparison of the classification performance of the earthquake dataset. The 1D CNN model produced the highest classification performance with the fastText word embedding technique, and the lowest was the BERT model.

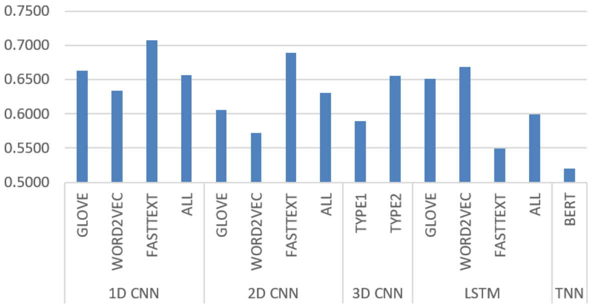

Figure 5: Comparison of performance on earthquake messages classification.

Figure 6 shows a comparison of the classification performance of flood datasets. The CNN type 1 3D model produced the highest classification performance and the lowest was the performance of the LSTM model with the fastText word embedding technique.

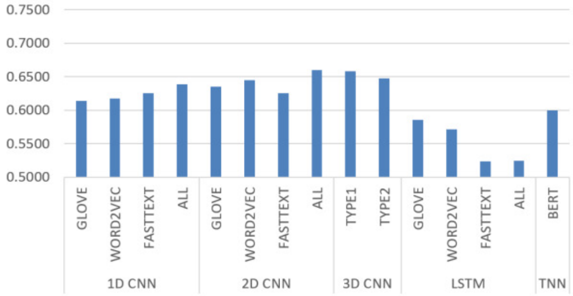

Figure 6: Comparison of performance on flood messages classification.

Figure 7 shows a comparison of the classification performance of forest fire datasets. The highest classification performance is produced by the 2D CNN model with the word embedding Word2Vec technique, and the lowest is the performance of the 1D CNN model with the combination of the three-word embedding techniques.

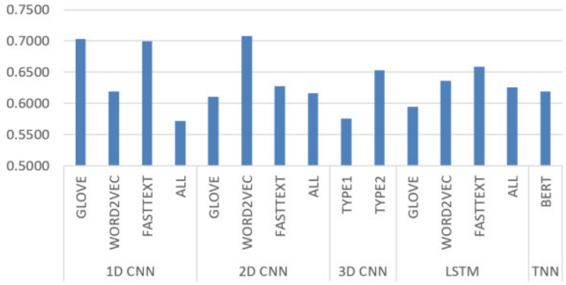

Figure 7: Comparison of performance on forest fire messages classification.

Figure 8 shows a comparison of the classification performance of hurricane datasets. The 3D CNN type 2 model produced the highest classification performance, and the lowest was the performance of the LSTM model with the word embedding Glove technique. The effect of increasing performance with feature extraction combining the three-word embedding techniques works well when used on 1D data with the 1D CNN classification method. This technique also improves the performance of the LSTM classification model.

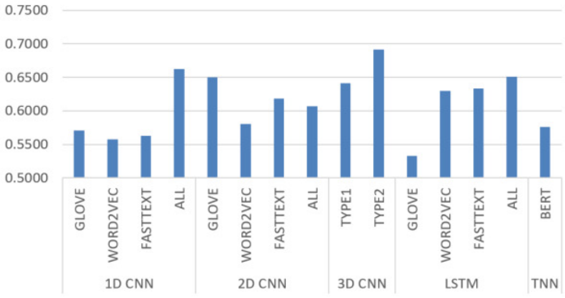

Figure 8: Comparison of performance on hurricane messages classification.

Figure 9 shows the average F1 score based on the classification method. This figure shows that the 3D CNN type 2 method as a feature extraction method proposed in this study, provides the highest average performance compared to other classification methods. In this research, BERT, a state-of-the-art classification method, cannot perform well in the case of natural disaster message classification.

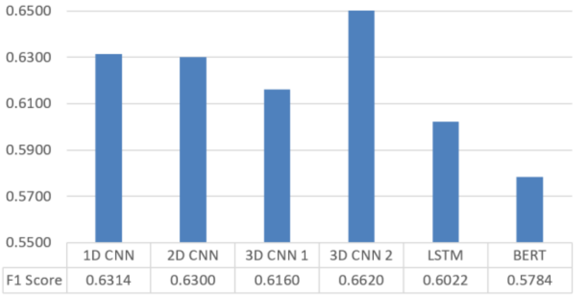

Figure 9. Average F1 Score by classification methods.

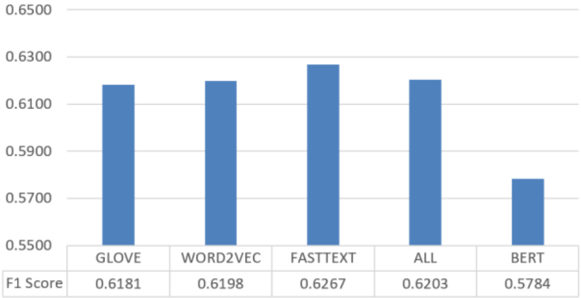

Figure 10: Average F1 score by word embedding methods.

Figure 10 shows the average F1 score based on the word embedding method. The highest classification performance is produced by the fastText-based feature extraction method. However, combining the three word embedding methods, the proposed feature extraction method can work better than the Glove and Word2Vec methods.

Further analysis of the performance of the 3D CNN type 2 model is to look at the predictive performance of each class in the dataset. Figure 11 shows the average AUC of each class using this model. This figure shows that the predictive performance of the eyewitness class is below the prediction performance of the other two classes.

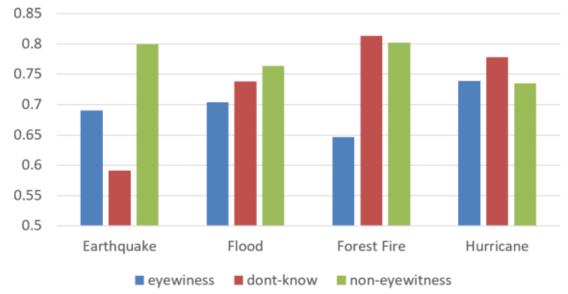

Figure 11: Average AUC of each class by 3D CNN Type 2.

The main reason for the low average performance of this prediction is because the number of messages in the eyewitness category is less than the messages in other categories, except for the earthquakes dataset. Another cause is that eyewitness category messages generally contain concise messages, namely 2-3 words, so the structured data that is formed includes a value of 0. Those reasons can impact the model training process and decrease classification performance [27].

Analysis of the prediction performance of the classes was also carried out by comparing the performance provided by the model formed by fastText-based feature extraction with the incorporation of three word embedding techniques. The results of the performance comparison can be seen in Figure 12. In this result it can be seen the decrease in the prediction performance of class eyewitness from combining the three word embedding techniques. Figure 12 also shows that the predictive performance of the eyewitness class is below the prediction performance of the other two classes. The reason of this issue is same as the explanation in previous paragraph.

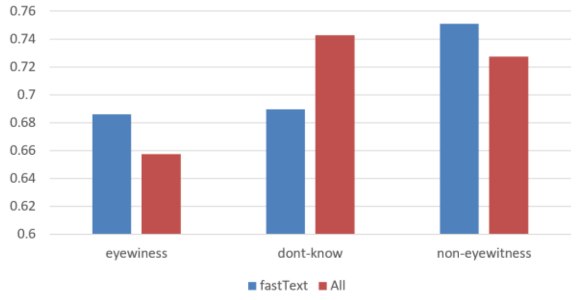

Figure 12: Performance comparison of classes prediction.

Conclusions

From the results of this study, it can be concluded that the formation of 3D data type 2 and 3D CNN models as the proposed method can provide a better average performance in the four cases of classification of natural disaster messages. In comparison, the proposed method combining three-word embedding techniques can improve classification performance in the 1D CNN and LSTM classification models.

However, the average performance for predicting messages from eyewitnesses is still lower than the predictive performance of other message categories. It is because the number of eyewitness category messages is less than the other categories, resulting in cases of unbalanced class classification, which decreases the performance of the minority class classification.

Future research will focus on solving unbalanced data classification cases by balancing the data before creating a classification model. This step is expected to improve the prediction performance for eyewitness category messages.

Acknowledgements

In this research, the computer system's computation time was provided by the Data Science Lab of the Computer Science Department, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University. This work was supported by the Program Dosen Wajib Meneliti (PDWM) grant from PNBP Lambung Mangkurat University.

References

[1] D. Wu, Y. Cui, Disaster early warning and damage assessment analysis using social media data and geo-location information, Decision Support Systems 111 (2018) 48-59, https://doi.org/10.1016/j.dss.2018.04.005.

[2] K. M Rodriguez, S. K. Ofori, L. C. Bayliss, J. S. Schwind, K. Diallo, M. Liu, J. Yin, G. Chowell, I. C. H. Fung, Social media use in emergency response to natural disasters: a systematic review with a public health perspective, Disaster Medicine and Public Health Preparedness, 14(1) (2020) 139-149, https://doi.org/10.1017/dmp.2020.3.

[3] K. Zahra, M. Imran, F. O. Ostermann, Automatic identification of eyewitness messages on twitter during disasters, Information Processing & Management, 57(1) (2020) 102-107, https://doi.org/10.1016/j.ipm.2019.102107.

[4] A. Devaraj, D. Murthy, A. Dontula, Machine learning methods for identifying social media-based requests for urgent help during hurricanes, International Journal of Disaster Risk Reduction 51 (2020) 101757, https://doi.org/10.1016/j.ijdrr.2020.101757.

[5] B. Jang, M. Kim, G. Harerimana, S. Kang, J. W. Kim, Bi-LSTM model to increase accuracy in text classification: Combining Word2vec CNN and attention mechanism, Applied Sciences 10(17) (2020) 5841, https://doi.org/10.3390/app10175841.

[6] M. R. Faisal, R. A. Nugroho, R. Ramadhani, F. Abadi, R. Herteno, T. H. Saragih, Natural Disaster on Twitter: Role of Feature Extraction Method of Word2Vec and Lexicon Based for Determining Direct Eyewitness, Trends in Sciences 18(23) (2021) 680-680, https://doi.org/10.48048/tis.2021.680.

[7] R. Rinaldi, M. R. Faisal, M. I. Mazdadi, R. A. Nugroho, F. Abadi, Eye witness message identification on forest fires disaster using convolutional neural network, Journal of Data Science and Software Engineering 2(02) (2021) 100-108.

[8] J. O. Luna, D. Ari, Word Embeddings and Deep Learning for Spanish Twitter Sentiment Analysis, Communications in Computer and Information Science, 898 (2019) 19-31, https://doi.org/10.1007/978-3-030-11680-4_4.

[9] D. Li, J. Zhang, Q .Zhang, X. Wei, Classification of ECG signals based on 1D convolution neural network, IEEE 19th International Conference on e-Health Networking Applications and Services (Healthcom) (2017) 1-6, https://doi.org/10.1109/HealthCom.2017.8210784.

[10] H.M. Rai, K. Chatterjee, 2D MRI image analysis and brain tumor detection using deep learning CNN model LeU-Net, Multimedia Tools and Applications 80 (2021) 36111–36141, https://doi.org/10.1007/s11042-021-11504-9.

[11] B. Khagi, G. R. Kwon, 3D CNN design for the classification of Alzheimer's disease using brain MRI and PET, IEEE Access 8 (2020) 217830-217847, https://doi.org/10.1109/ACCESS.2020.3040486.

[12] Y. Goldberg, O. Levy, word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method, arXiv:1402.3722 (2014).

[13] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (2014) 1532-1543.

[14] U. D. Gandhi, P. M. Kumar, G. C. Babu, G. Karthick, Sentiment Analysis on Twitter Data by Using Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM), Wireless Personal Communications (2021) 1-10, https://doi.org/10.1007/s11277-021-08580-3.

[15] A. Bhoi, S. P. Pujari, R. C. Balabantaray, A deep learning-based social media text analysis framework for disaster resource management, Social Network Analysis and Mining 10(78) (2020) 1-14, https://doi.org/10.1007/s13278-020-00692-1.

[16] J. Devlin, M. W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv:1810.04805 (2018).

[17] S. G. Carvajal, E. C. G. Merchán, Comparing BERT against traditional machine learning text classification. arXiv:2005.13012 (2020).

[18] W. Maharani, Sentiment analysis during Jakarta flood for emergency responses and situational awareness in disaster management using BERT, 8th International Conference on Information and Communication Technology (ICoICT) (2020) 1-5, https://doi.org/10.1109/ICoICT49345.2020.9166407.

[19] M. K. Delimayanti, R. Sari, M. Laya, M. R. Faisal, R. F. Naryanto, The effect of pre-processing on the classification of twitter's flood disaster messages using support vector machine algorithm, 3rd International Conference on Applied Engineering (ICAE) (2020) 1-6, https://doi.org/10.1109/ICAE50557.2020.9350387.

[20] S. Khomsah, R. D. Ramadhani, S. Wijaya, The Accuracy Comparison Between Word2Vec and FastText On Sentiment Analysis of Hotel Reviews, Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi) 6(3) (2022) 352-358, https://doi.org/10.29207/resti.v6i3.3711.

[21] F. Anistya, E. B. Setiawan, Hate Speech Detection on Twitter in Indonesia with Feature Expansion Using GloVe, Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi) 5(6) (2021) 1044-1051, https://doi.org/10.29207/resti.v5i6.3521.

[22] R. Chauhan, K. K. Ghanshala, R. C. Joshi, Convolutional neural network (CNN) for image detection and recognition, 1st International Conference on Secure Cyber Computing and Communication (ICSCCC) (2018) 278-282, https://doi.org/10.1109/ICSCCC.2018.8703316.

[23] A. K. Dubey, V. Jain, Comparative Study of Convolution Neural Network's Relu and Leaky-Relu Activation Functions, Applications of Computing, Automation and Wireless Systems in Electrical Engineering (2019) 873-880 https://doi.org/10.1007/978-981-13-6772-4_76.

[24] S. S. Basha, S. R. Dubey, V. Pulabaigari, S. Mukherjee, Impact of fully connected layers on performance of convolutional neural networks for image classification, Neurocomputing 378 (2020) 112-119, https://doi.org/10.1016/j.neucom.2019.10.008.

[25] S. Bodapati, H. Bandarupally, R.N. Shaw, A. Ghosh, Comparison and Analysis of RNN-LSTMs and CNNs for Social Reviews Classification, Advances in Applications of Data-Driven Computing (2021) 49-59, https://doi.org/10.1007/978-981-33-6919-1_4.

[26] Y. Yu, X. Si, C. Hu, J. Zhang, A review of recurrent neural networks: LSTM cells and network architectures, Neural Computation 31(7) (2019) 1235-1270, https://doi.org/10.1162/neco_a_01199.

[27] J. Bissmark, O. Wärnling, The Sparse Data Problem Within Classification Algorithms : The Effect of Sparse Data on the Naïve Bayes Algorithm (Dissertation), (2017). http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-209227.