International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 14, No.1 (Sep-2023)

http://dx.doi.org/10.12785/ijcds/140174

Disaster Tweet Classifications Using Hybrid Convolutional Layers and Gated Recurrent Unit

Ricko Anugrah Mulya Pratama[1] **and Hilman Ferdinandus Pardede**[1,2]

[1]*Graduate School of Computer Science, Faculty of Information Technology, Nusa Mandiri University, Jakarta, Indonesia*
[1,2]*Research Center for Artificial Intelligence and Cyber Security, National Research and Innovation Agency, Bandung, Indonesia*

Received 02 Mar. 2023, Revised 22 May 2023, Accepted 31 Jul. 2023, Published 01 Sep. 2023

Abstract: A disaster monitoring system using Twitter data can provide information regarding disaster-prone areas and emergency response information. There have been several studies aimed at applying machine learning technologies to automatically detect disaster information from Twitter data. The Support Vector Machine (SVM) is one of the frequently used algorithms for text categorization situations, but SVM for text classification is limited by drawbacks transparency in the results caused by the high number of dimensions. Long Short-Term Memory (LSTM) is another deep learning technique that is frequently employed for text categorization, but the LSTM processing process uses quite long stages so that it requires longer computation time. The main idea in proposing this hybrid model is to combine the advantages of a highly reliable Convolutional Neural Network (CNN) architecture for handling high-dimensional data and Gated Recurrent Units (GRU) which are effective in processing sequential data and have faster computation time compared to LSTM. This study uses NLP Disaster Tweets dataset from Kaggle. The suggested model outperforms at least 12 different categories of conventional machine learning algorithms as well as other widely used deep learning models in terms of performance. The CNN-GRU hybrid model with FastText produces an accuracy of 83.32%, F1-score of 81.45%, and an AUC of 83.45%.

Keywords: Disaster Tweets, Classification, Hybrid CNN-GRU, Deep Learning, Natural Language Processing

1. INTRODUCTION

Twitter is a platform where opinions can be obtained on almost every subject. One of the most widely obtained information from Twitter is news about disasters, which allows the public to share disaster information in real time with the public [1]. A disaster monitoring system using Twitter data can provide information regarding disaster-prone areas and emergency response information.

The occurrence of disasters is not only caused by natural disasters, but also many are caused by human negligence. The effect of a catastrophe can include human casualties, property loss, social and economic disruption, and environmental harm, according to the United Nations International Strategy for catastrophe Reduction (UNISDR) [2].

Because of this, a lot of news organizations and disaster aid groups are interested in automating the monitoring of catastrophe information on Twitter. To make this happen, a machine learning algorithm is needed that can automatically identify text from Twitter that can recognize disaster-related contexts or not.

The main idea of this research is to create a machine learning model that can analyze text on Twitter that refers to disasters (fires, earthquakes, etc.), to help mobilize emergency response teams quickly. The challenge in this research is that every tweet has a pattern, long sentences, and content that is not just content text data, but can also contain a wide variety of photos, videos, or web links.

The next challenge is how well machine learning models can classify a valid disaster tweet, just humor, or just a metaphor. Computer algorithms cannot recognize raw text, Natural Language Processing (NLP) is needed to help computers understand natural human language [3]. Thusly, machine learning can learn patterns of textual data.

The classification case encountered in this study is binary classification, the model is designed to sort out tweet information related to disaster or not. In the case of binary classification, many studies have tried different types of classifiers. One of the commonly used classifiers is Support Vector Machine (SVM).

SVM has been widely used for various text binary classification studies, because SVM by nature is binary classifier [4]. SVM is very reliable in finding the best hyperplane by maximizing the distance between classes. However, SVM for text classification is limited by drawbacks transparency in the results caused by the high number of dimensions [5].

In addition, the deep learning approach that is commonly used for text classification is Long Short-Term Memory (LSTM). LSTM is able to learn and considering long-term dependability [6]. However, LSTM uses long stages so it requires a longer computation time [7]. This computational

E-mail: 14207093@nusamandiri.ac.id, hilm003@brin.go.id

problem is certainly very time consuming, especially for handling large amounts of tweet data.

The main idea in proposing this hybrid model is to combine the advantages of a highly reliable Convolutional Neural Network (CNN) architecture for handling high-dimensional data and extracting local features from text [8]. The advantages of CNN are combined with the Gated Recurrent Unit (GRU) which can process sequence data very well and has a faster computation time compared to LSTM [9]. With each of these advantages, the combination is expected to produce an optimal classifier model, so that it can contribute to the development of an early warning system for disaster detection.

2. LITERATURE REVIEW

Regarding text classification, there have been many studies in the field of NLP that utilize social media data, especially twitter for various research needs. In 2019 [10], a study presents a methodology for identifying fake news using a mixture of Naïve Bayes classifiers, Support Vector Machines. The suggested model is effective at defining the accuracy of outcomes.

The Naive Bayes approach was applied in a different research in 2021 [11], for Twitter Sentiment Analysis of COVID-19 Vaccines in the Philippines. The Naive Bayes method was used to annotate and train sentiments in order to classify English and Filipino tweets into positive, neutral, and negative attitudes. This algorithm does well even with a small dataset made up of tweets from the first month of the Philippines vaccination program.

Similar studies have been carried out, the authors used Twitter data to classify hate speech using the Multinomial Logistic Regression Method [12]. Based on the results of this study can be drawn the conclusion that the process of classifying hate speech by using the Multinomial Logistic Regression capable of producing a good classification.

The ensemble classification technique has also been applied in several studies [13], [14] for example to develop a detection model for ironic Arabic tweets. This work uses a simple TF/IDF with n-gram model range to extract features to train the classifier. Ensemble-based classifier gives better F1-score than another basic classifier.

There is research that does not only focus on algorithms, but also measures the performance of word embedding techniques using Global Vector (GloVe), FastText and Word2Vec [15]. Each was tested using CNN for text classification. Based on the test results, FastText provides the best accuracy performance compared to Glove and Word2vec which were tested using 20 newsgroup datasets.

A study published in 2020 [16] examined the use of hostile and combative language in YouTube comments. This study compares two text representations, bag of words (BOW) and pretrained word embedding, using a binary classification strategy for predicting threats that are directed at specific persons vs groups. The lowest performance comes from GloVe, FastText performs significantly better, and the highest performance comes from Bidirectional LSTM with a TF-IDF weighting scheme.

In 2021 [17], for the purpose of detecting hate speech in Indonesian Twitter texts, a bidirectional GRU (BiGRU) approach has been created. Additionally, this work uses Word2vec and fastText to achieve word embedding. Both of these word embedding techniques produce similar accuracy. The two methods that this study suggests are BiGRU with trained IndoBERT and BiGRU without any prior training on other models. The classification method with the highest accuracy uses IndoBERT with Gated Recurrent Units without Stop Word Removal.

Hybrid methods on deep learning approaches are also applied by some researchers for text classification [18]. The study entitled "A Hybrid CNN-LSTM Model for Psychopathic Class Detection from twitter Users". The index of each word acquired in the preceding phase was converted in this study using the random word embedding approach into a fixed-size real value vector. This research uses techniques based on hybrid CNN-LSTM model. The proposed hybrid model produces better accuracy than the classical machine learning model.

3. PROPOSED METHODOLOGY

In this study, to determine which tweets were a complete catastrophe and which weren't, we examined a dataset of 10,876 given tweets. The figure eight firm produced this dataset, which was first made available on their website. Until this research was made, the dataset we used was still being used for competitions on Kaggle [19].

Our issue may be resolved as a binary classification task, with the desired result being the label y = 1 if the tweet is about a true tragedy and y = 0 otherwise. The following are some of the steps taken in this research:

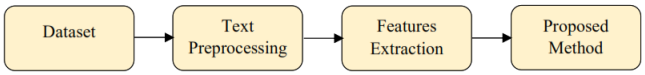

Figure 1. Workflow of Proposed Methodology

Twitter data is closely related to sharing links, hashtags to highlight a topic, the @ sign to refer to other users, the use of emoticon character symbols, word abbreviations, and irregular sentence structures. The characteristics of text on Twitter should be studied further before going to the word embedding stage. One important element of many text mining techniques is text preparation. Tasks like tokenization, filtering, lemmatization, and stemming are typically included. Figure 2. below depicts the steps we took to preprocess the text.

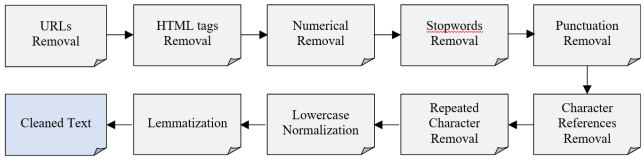

Figure 2. Stages of Text Preprocessing

The first step we take is to remove some of the information that is not needed for text identification. In between,

URLs character references (e.g., https, http), remove HTML tags (e.g., /p, tbody, h1, pre), remove numeric values including mixtures of alphabetical characters (e.g., 5km, M194), remove stopwords, remove mentions in tweets, and remove punctuation.The next step, remove character references (e.g., lt;, amp;, nbsp;), remove repeated charcters in elongated words, then we uniformize the entire text to lowercase, and the last step is change a word into basic tenses by knowing the context of the word also known as lemmatization.

We present a five-row example from a dataset for comparing text before and after text preprocessing. All text processing functions have been running well, every word has been converted to lower case. On the first line to the third line it looks like the stopwords removal is going well. Likewise, unnecessary punctuation such as hashtags are also removed. The lemmatization technique also seems to work, all sentences in the dataset have been successfully converted into basic forms of tenses. The results of text pre-processing can be seen as follows:

TABLE I. TEXT PREPROCESSING RESULT

Id	Raw text	Cleaned text
0	Just happened a terrible car crash	*happen terrible car crash*
2	Heard about #earthquake is different cities, stay safe everyone.	*hear earthquake different city stay safe*
3	There is a forest fire at spot pond, geese are fleeing across the street, I cannot save them all.	*forest fire spot pond geese flee street save*
9	Apocalypse lightning. #Spokane #wildfires	*apocalypse lightning spokane wildfire*
11	Typhoon Soudelor kills 28 in China and Taiwan	*typhoon soudelor kill china taiwan*

The next step is the feature extraction process, this process can provide additional information and facilitate learning in the guided learning model by converting text data into numerical vector representations. The application of the word embedding technique can produce vector values for each word from the data corpus so that it can represent the relationship of one word to another.

The initial process at this stage is to vectorize the text that has gone through the preprocessing stage, the next process is to convert each text into a sequence of numbers (each number becomes a tokenization index in the dictionary). The result is an ordered list of variable sizes, this process is necessary because the word length of each tweet varies widely. Therefore, the Pad sequence function is needed to normalize sentences in tweet data with uniform sentence lengths [20].

The first step is to obtain a tweet with the maximum word count, after which the padding procedure is carried out by the pad sequence using the post function. Padding post serves to add the number 0 behind the matrix whose number of words is less than the maximum number of words that have been processed before. Pad sequence result example:

```
array([[3552,  395,  169, ...,    0,    0,    0],
       [ 110,    2,  149, ...,    0,    0,    0],
       [1345,  445, 1494, ...,    0,    0,    0],
       ...,
       [1218,   25,  404, ...,    0,    0,    0],
       [  20,  723,  393, ...,    0,    0,    0],
       [ 116,  647,  359, ...,    0,    0,    0]])
```

Figure 3. Pad Sequences Results

The next stage is the implementation of pre-trained word embedding using Global Vector (GloVe) and FastText. The use of two types of word embedding aims to enrich the performance information from the proposed CNN-GRU hybrid model. So that we can find out in more detail how far the word embedding influences the performance of the model. Both do the same task but use different strategies to convert text into vector sets. The embedding dimension used is 300d respectively and the training data on clean tweets is 7.613. Here's an example:

```
0       [-0.5986820042133332, 0.036138801276683806, -0...
1       [-0.01481742677944047, -0.09212342649698257, -...
2       [0.06581491028720682, 0.047786547379060226, -0...
3       [-0.17240000267823538, 0.23847616898516813, -0...
4       [-0.1398800015449524, -0.07596778372923534, -0...
7608    [0.02860242554119655, -0.15843028042997634, -0...
7609    [-0.17457243161542074, 0.11100599808352334, 0....
7610    [0.048529799282550815, 0.024588394165039062, 0...
7611    [-0.06087812455371022, 0.05221772601362318, -0...
7612    [-0.07772537937853485, 0.2898400016129017, -0....
Name: text_cleaned, Length: 7613, dtype: object
```

Figure 4. Word Embedding Results using GloVe

```
0       [-0.021920000575482844, -0.011399999796412884,...
1       [-0.029514286351124092, 0.014042856271511741, ...
2       [0.008100000040774996, -0.02130909088406373, 0...
3       [0.016666666449358065, -0.0474666675630336, 0....
4       [-0.009455555429061254, -0.0543333324086335, 0...
7608    [-0.01735714197690998, -0.024014286563864777, ...
7609    [0.002371428567650063, -0.017414286067443236, ...
7610    [0.032980003580451014, 0.02460000109858811, 0....
7611    [0.006387499219272286, 0.009787500071979593, 0...
7612    [-0.020612499909475446, -0.015887500368990004,...
Name: text_cleaned, Length: 7613, dtype: object
```

Figure 5. Word Embedding Results using FastText

The results of the word embedding shows the same 5 initial data rows and 5 final data rows, it can be seen that there are differences in the resulting vector shape between FastText and GloVe. To determine the semantic links between words in the corpus, GloVe uses word or co-occurrence information from global statistics [21]. GloVe employs the global matrix factorization technique. FastText, on the other hand, learns word representation while taking subword information into consideration [22]. Each word has a unique collection of n-gram characters to represent it.

After the word embedding process, the next step is to prepare the classifier model architecture. This CNN-GRU hybrid model's major goal is to bring together the strengths of the GRU module, which is dependable when handling sequence data, and the superior CNN module for handling high-dimensional data. Here is the proposed architecture:

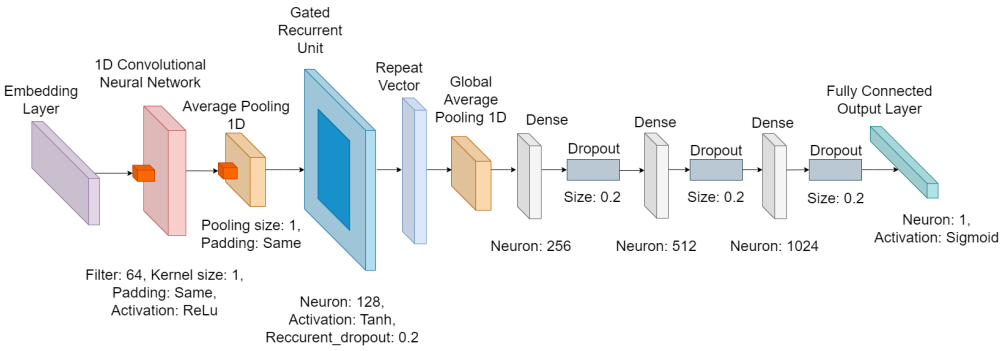

Figure 6. Proposed Hybrid CNN-GRU Architecture

TABLE II. PROPOSED HYBRID CNN-GRU MODEL SUMMARY

Layer (type)	Output Shape	Parameters
Embedding	(None, 22, 300)	3.644.400
Convolutional 1D	(None, 22, 64)	19.264
Average Pooling 1D	(None, 22, 64)	0
Gated Recurrent Unit	(None, 128)	74.496
Repeat Vector	(None, 22, 128)	0
Global Average Pooling 1D	(None, 128)	0
Dense	(None, 256)	33.024
Drop Out	(None, 256)	0
Dense	(None, 512)	131.584
Drop Out	(None, 512)	0
Dense	(None,1024)	525.312
Drop Out	(None, 1024)	0
Dense (Fully Connected)	(None, 1)	1025
Total Parameters		***4.429.105***
Trainable Parameters		***784.705***
Non-Trainable Parameters		***3.644.400***

Embedding is the first layer that captures the vector representation of the text. Then the convolution process is carried out using 1D CNN with a total of 64 filters, kernel size 1, the same padding and using ReLu activation. The 1D CNN output is then subjected to an Average Pooling layer, which reduces its dimensionality by computing the average number. The average pooling output is forwarded to the GRU layer with a total of 128 neurons.

At the GRU layer, we use 0.2 recurrent dropouts and tanh activation. Next, we use the RepeatVector layer to convert the 2d shape of the GRU layer to a 3d shape and pass it to the GlobalMaxPooling1D layer to restore the 2d shape. A dense layer with neuron configuration twice as large as the preceding layer will receive the output from the GlobalMaxPooling1D layer, followed by a dropout layer with a size of 0.2.

Then the combination of dense layers and dropouts is repeated three times in a row, the only difference in configuration is the size of the number of neurons in each dense layer. The configuration of one neuron's value and the sigmoid activation function at the fully connected layer are provided to the final layer, which decides whether to put the tweet in the disaster category or not.

The summary table shows the order of layers in the hybrid architecture used with a total of 13 layers. The resulting 3d shape output consists of four layers including the Embedding, 1D Convolutional, AveragePooling1D layers and the last is the RepeatVector layer, the remaining 9 layers form the 2d shape output. The total parameters generated for training data are recorded as more than 4.4 million parameters with a proportion of only around 18% or 784.705 parameters that will become training data.

4. RESULT AND DISCUSSION

The configuration uses a test proportion of 20%, epoch used is 10, learning rate used is 0.001 followed by the auto reduce learning rate function to optimize loss validation during the training process. When the validation loss does not increase from the previous epoch, This feature will be turned on automatically and will lower the previous learning rate's size. All deep learning models employ the Adam optimizer, and the batch size is 32. Figure below shows the training history of our proposed CNN-GRU hybrid model.

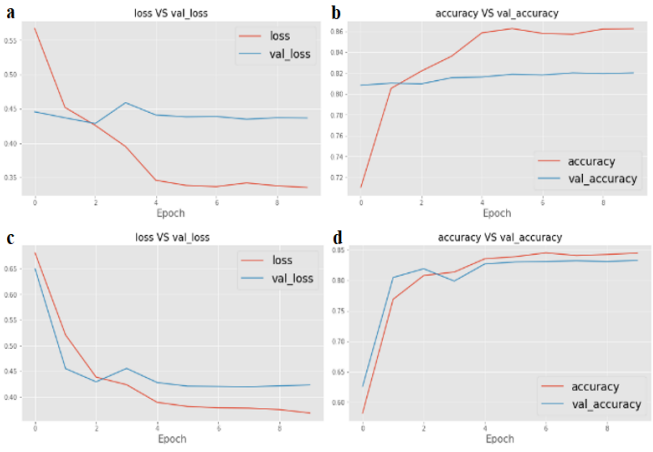

Figure 7. Graph of Training History
(a & b) GloVe; (c & d) FastText;

The training graph shows the proposed CNN-GRU hybrid model works fine when using both types of word embedding. However, when viewed from the validation line pattern on the resulting FastText graph, it looks more stable than GloVe. The graphics generated by GloVe look out of place, where there is a huge gap between their respective accuracy, loss and validation. When the CNN-GRU hybrid model uses GloVe, the experimental results show that loss validation does not continue from the 4th to the 10th epoch.

When the hybrid CNN-GRU model uses FastText, there is a notable difference in performance. This can be seen from the loss validation which did not increase only in epochs 4, 9 and 10. However, the difference between the score and the validation is not too far away. FastText performs better based on validation scores that are stable enough to follow a line pattern during training.

The following is a table of classification results that we compare based on the word embedding technique used. The score in bold aims to find out the highest score in each metric category used. The suggested CNN-GRU hybrid model, as shown in the two tables below, has the best accuracy, f1-score, and AUC scores overall when compared to 12 classical machine learning classifiers and 3 other types of deep learning models. The hybrid model was also tested in combination with CNN-LSTM, but the CNN-GRU model still outperformed all types of applied classifiers.

TABLE III. CLASSIFICATION RESULTS USING GLOVE

Classifiers	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC Score (%)
Decision Tree	64.34	62.42	58.81	60.56	63.98
K-NN	75.96	72.01	79.12	75.40	76.17
AdaBoost	77.15	76.42	73.62	75.00	76.92
Naive Bayes	77.80	74.31	79.97	77.03	77.94
Ridge	78.59	77.87	75.45	76.64	78.39
Random Forest	78.72	84.19	66.85	74.52	77.96
XGBoost	78.85	79.09	74.18	76.56	78.55
Logistic Regression	79.12	77.88	77.00	77.44	78.98
SVM	79.51	79.58	75.31	77.39	79.24
ExtraTrees	79.77	**86.12**	67.41	75.63	78.97
Gradient Boost	80.36	82.53	73.34	77.66	79.91
CatBoost	80.69	81.87	75.17	78.38	80.33
1D CNN	80.95	77.43	80.85	79.10	80.94
LSTM	81.74	76.72	**82.80**	79.64	81.87
GRU	81.81	77.57	82.33	79.88	81.86
CNN-LSTM	81.87	77.29	82.65	79.88	81.96
CNN-GRU	**82.00**	78.13	82.31	**80.17**	**82.04**

TABLE IV. CLASSIFICATION RESULTS USING FASTTEXT

Classifiers	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC Score (%)
Decision Tree	63.95	61.14	61.91	61.52	63.82
Naive Bayes	65.79	60.28	77.71	67.89	66.56
K-NN	72.68	67.41	79.97	73.16	73.15
AdaBoost	76.82	76.96	71.65	74.21	76.48
Random Forest	78.06	84.91	64.31	73.19	77.18
SVM	78.85	79.63	73.34	76.35	78.50
ExtraTrees	79.25	**86.72**	65.44	74.59	78.36
Gradient Boost	79.84	81.01	74.04	77.37	79.46
Ridge	80.10	80.94	74.89	77.80	79.76
Logistic Regression	80.23	82.58	72.91	77.45	79.76
XGBoost	80.43	81.86	74.47	77.99	80.04
CatBoost	81.41	83.17	75.31	79.05	81.02
1D CNN	81.54	77.85	81.65	79.71	81.56
GRU	81.61	74.18	84.43	78.97	82.04
LSTM	81.87	72.63	**86.26**	78.86	82.65
CNN-LSTM	82.07	76.30	83.74	79.85	82.29
CNN-GRU	**83.32**	78.70	84.41	**81.45**	**83.45**

Both tables are sorted from lowest to highest accuracy scores. Both tables show consistency that the ExtraTrees classifier has the highest precision score, while the highest recall score is achieved consistently by LSTM. Among all classifiers, decision tree has the lowest accuracy score, this is proven consistently when using GloVe and FastText. Meanwhile, the CatBoost classifier is the only classifier whose accuracy can approach deep learning algorithms. Although ExtraTrees and LSTM have the highest precision and recall scores respectively, these results do not yet determine which model is the best.

However, the final model performance results evaluated using the F1-Score. The F1-score metric complies with the Kaggle competition rules listed in the dataset we used. In essence, the F1-score combines the precision and recall ratings by averaging them. In addition, we also add AUC Score to enrich the results of measurements.

Overall, it is clear that the proposed CNN-GRU hybrid architecture, even when evaluated using two separate pre-trained word embedding methodologies, can deliver classification performance that is compatible with obtaining the highest possible score. To find out the details of true and false classification in the proposed hybrid model, see the confusion matrix. The confusion matrix below shows information for true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN).

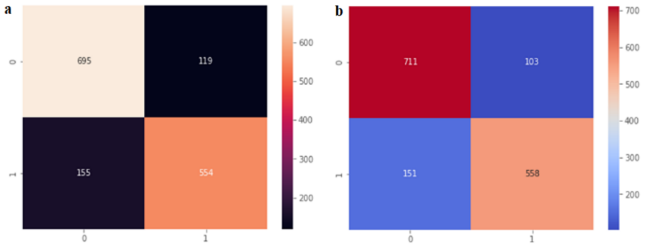

Figure 8. Confusion Matrix of the Proposed Model
(a) GloVe; (b) FastText;

TABLE V. COMPARISON OF CLASSIFICATION RESULTS

Category	Number of tweets	
	GloVe	FastText
True Positive (0)	695	711
True Negative (1)	554	558
False Positive	119	103
False Negative	155	151

The confusion matrix's findings demonstrate that the suggested Hybrid CNN-GRU model performs more optimally while utilizing FastText for classification. CNN-GRU hybrid model performance using FastText resulted in a difference in the number of correct classifications reaching 16 tweets non disaster category (0) and 4 disaster category tweets (1). Then, by number of correct classification differences up to 20 tweets for comparison using the GloVe word embedding technique. The ROC metrics below also show satisfactory results when the hybrid model is combined with FastText, the difference in scores reaches 1.5%.

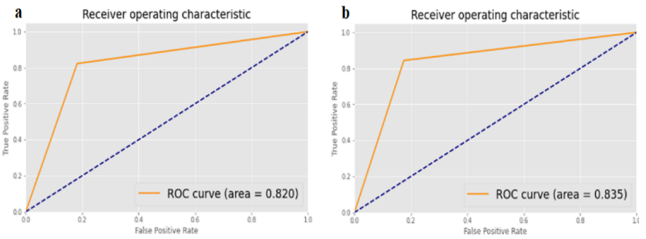

Figure 9. ROC Curve of the Proposed Model
(a) GloVe; (b) FastText;

5. CONCLUSIONS

This research tries many classifiers to ensure our proposed model is right to solve the problem of classifying disaster tweets. Both of the pre-trained word embedding methods were utilized to assess how well our suggested model performed. When compared to several other classifiers, all tests consistently demonstrate that our suggested model has the greatest classification performance, especially based on accuracy, f1-score and AUC score. An accuracy score of 83.32%, an F1-score of 81.45%, and an AUC score of 83.45% may be achieved using our hybrid CNN-GRU model with FastText.

Results of the comparison demonstrate that FastText performs somewhat better when combined with our proposed hybrid CNN-GRU for the dataset cases we are working on. With these results, we have succeeded in proving that the performance of our proposed hybrid model has good consistency when adapting two different word embedding techniques. However, with other dataset the results may be different. To be sure, this CNN-GRU hybrid model needs to be tested with more text-based datasets in future work.

REFERENCES

[1] Z. Ashktorab, C. Brown, M. Nandi, and A. Culotta, "Tweedr: Mining twitter to inform disaster response." *ISCRAM*, pp. 269–272, 2014.

[2] U. N. O. for Disaster Risk Reduction (UNISDR), "2009 unisdr terminology on disaster risk reduction," 2009, https://www.unisdr.org/files/7817_UNISDRTerminologyEnglish.pdf [Accessed: (20 April 2023)].

[3] C. V. B. Murthy, M. L. Shri, S. Kadry, and S. Lim, "A survey of multilingual neural machine translation," *IEEE Access*, vol. 8, pp. 205 190–205 205, 2020.

[4] S. Chatterjee, P. G. Jose, and D. Datta, "Text classification using svm enhanced by multithreading and cuda," *International Journal of Modern Education and Computer Science*, vol. 12, no. 1, p. 11, 2019.

[5] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, 2019.

[6] A. Kumar, V. T. Narapareddy, V. A. Srikanth, A. Malapati, and L. B. M. Neti, "Sarcasm detection using multi-head attention based bidirectional lstm," *Ieee Access*, vol. 8, pp. 6388–6397, 2020.

[7] Y. Zhang, Q. Liu, and L. Song, "Sentence-state lstm for text representation," *arXiv preprint arXiv:1805.02474*, 2018.

[8] R. Wang, Z. Li, J. Cao, T. Chen, and L. Wang, "Convolutional recurrent neural networks for text classification," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–6.

[9] W. Wu, W. Liao, J. Miao, and G. Du, "Using gated recurrent unit network to forecast short-term load considering impact of electricity price," *Energy Procedia*, vol. 158, pp. 3369–3374, 2019.

[10] A. Jain, A. Shakya, H. Khatter, and A. K. Gupta, "A smart system for fake news detection using machine learning," in *2019 International conference on issues and challenges in intelligent computing techniques (ICICT)*, vol. 1. IEEE, 2019, pp. 1–4.

[11] C. Villavicencio, J. J. Macrohon, X. A. Inbaraj, J.-H. Jeng, and J.-G. Hsieh, "Twitter sentiment analysis towards covid-19 vaccines in the philippines using naïve bayes," *Information*, vol. 12, no. 5, p. 204, 2021.

[12] P. S. B. Ginting, B. Irawan, and C. Setianingsih, "Hate speech detection on twitter using multinomial logistic regression classification method," in *2019 IEEE International Conference on Internet of*

Things and Intelligence System (IoTaIS). IEEE, 2019, pp. 105–111.

[13] H. A. Nayel, W. Medhat, and M. Rashad, "Benha@ idat: Improving irony detection in arabic tweets using ensemble approach." in *FIRE (Working Notes)*, 2019, pp. 401–408.

[14] S. Hakak, M. Alazab, S. Khan, T. R. Gadekallu, P. K. R. Maddikunta, and W. Z. Khan, "An ensemble machine learning approach through effective feature extraction to classify fake news," *Future Generation Computer Systems*, vol. 117, pp. 47–58, 2021.

[15] E. M. Dharma, F. L. Gaol, H. Warnars, and B. Soewito, "The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification," *Journal of Theoretical and Applied Information Technology*, vol. 100, no. 2, p. 31, 2022.

[16] N. Ashraf, R. Mustafa, G. Sidorov, and A. Gelbukh, "Individual vs. group violent threats classification in online discussions," in *Companion Proceedings of the Web Conference 2020*, 2020, pp. 629–633.

[17] A. Marpaung, R. Rismala, and H. Nurrahmi, "Hate speech detection in indonesian twitter texts using bidirectional gated recurrent unit," in *2021 13th International Conference on Knowledge and Smart Technology (KST)*. IEEE, 2021, pp. 186–190.

[18] P. K. Jain, V. Saravanan, and R. Pamula, "A hybrid cnn-lstm: A deep learning approach for consumer sentiment analysis using qualitative user-generated contents," *Transactions on Asian and Low-Resource Language Information Processing*, vol. 20, no. 5, pp. 1–15, 2021.

[19] Kaggle, "Natural language processing with disaster tweets," 2023, https://www.kaggle.com/competitions/nlp-getting-started [Accessed: (20 April 2023)].

[20] M. Ihsan, B. S. Negara, and S. Agustian, "Lstm (long short term memory) for sentiment covid-19 vaccine classification on twitter," *Digital Zone: Jurnal Teknologi Informasi Dan Komunikasi*, vol. 13, no. 1, pp. 79–89, 2022.

[21] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[22] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," *arXiv preprint arXiv:1712.09405*, 2017.

Ricko Anugrah Mulya Pratama obtained an Associate degree in Computer Engineering from Bina Sarana Informatika University in 2014. His Bachelor's Degree in Informatics Engineering and a Master's degree in Computer Science from Nusa Mandiri University in 2016 and 2023 respectively. His research interests are artificial intelligence, deep learning, big data, natural language processing, and face recognition.

Hilman Ferdinandus Pardede is research professor at Indonesian National Research and Innovation Agency, Research Center for AI and Cybersecurity. He obtained his bachelor degree in Electrical Engineering from University of Indonesia in 2004. His master degree in Information and Communication Technology were from The University of Western Australia in 2009. He obtained his Doctoral degree in Computer Science from Tokyo Institute of Technology in 2013. His research interests are speech and signal processings, machine learning and pattern recognition, and artificial intelligence.