Predicting Disasters from Tweets Using GloVe Embeddings and BERT Layer Classification

Aabha Ranade[1(✉)], Saurav Telge[1(✉)], and Yash Mate[2(✉)]

[1] B.E. Computer Engineering, Vivekanand Education Society's Institute of Technology, Mumbai, India
{2018.aabha.ranade, 2018.saurav.telge}@ves.ac.in
[2] MS Computer Science, University of Southern California, California, USA
ymate@usc.edu

Abstract. Twitter, a social media platform, has quickly become one of the most reliable sources of news and other information. With an ever-increasing number of users and tweets, it's feasible to use Twitter data to learn about a variety of interesting things that are happening around us. During a disaster, people can get real-time information from Twitter to give and receive help. This paper deals with the extraction of Twitter data in order to identify the tweets that give information about disasters. This is essentially a binary classification problem. GloVe Global Vectors for Word Representation embeddings have been implemented to convert the tweets into vectors which are then trained using the BERT (Bidirectional Encoder Representations from Transformers) model to classify the tweets into two categories: tweets related to disasters and tweets unrelated to disasters. This can be helpful to know how many tweets are related to disasters and are truly informative. Existing research focus on using LSTM, classification models (Random forest, Decision trees, Naive Bayes) which do not give accurate results. The results obtained in the proposed solution got an accuracy of around 87% in both training and validation parts. Thus, the BERT model is better as compared to other models.

Keywords: Disaster prediction · GloVe embeddings · BERT classification · Tweets classification · NLP

1 Introduction

As of 2021, Twitter is one of the most used social media platforms with approximately 200 million daily active users and over 500 million tweets sent per day [1]. This provides an enormous opportunity for data mining and analysis of tweets.

During a crisis, this medium is seen as an excellent place for harvesting information to determine what is happening on the ground. The increased usage of social media, especially during crises provides new information sources from which the relevant authorities can get better insights into the situation under consideration, which is commonly regarded as a vital component of making successful and effective emergency response decisions. Exploiting and analyzing disaster-related tweets is critical since the knowledge gleaned can be utilized to improve disaster response. The subject

D. Garg et al. (Eds.): IACC 2021, CCIS 1528, pp. 492–503, 2022.
https://doi.org/10.1007/978-3-030-95502-1_37

and content of tweets vary greatly, and the influx of tweets, especially in the aftermath of a catastrophe, can be overwhelming. It consists of socio-behavioral patterns such as intensified information seeking and increased information dissemination. Using the Twitter Search APIs [2], NodeXL [3] and other such tools, data from Twitter can be retrieved with the help of Twitter hashtags. The tweets can then be categorized as relevant (related to disasters) or irrelevant. There are three main approaches to this issue: Filtering tweets based on factors such as location and the presence of keywords or hashtags, crowdsourcing, and machine-learning approaches.

This paper presents a novel approach for predicting disasters through Tweets. We have performed data analysis on a dataset from a Kaggle competition called Real or Not? NLP with Disaster Tweets [4], which consists of training and testing sets having the text of the tweets, keywords present in the tweets, and the location the tweet was sent from. The task is to predict whether a given tweet is about a real disaster or not. To address this text classification problem we have used word embedding transformation followed by a BERT model. Word embedding is the representation of text data into numerical vector format which can be given as an input to machine learning models. In this project, we have made use of Stanford's GloVe (Global Vectors for Word Representation) embeddings. Next, we have employed a pre-trained BERT model to perform the classification. The performance of the model is tested by calculating the precision, recall, F1 score, and accuracy.

2 Literature Survey

2.1 Text Classification Using Machine Learning Techniques

Authors M. Ikonomakis, S. Kotsiantis, and V. Tampakas illustrate the text classification process using machine learning techniques in their paper. The major goal of this work is to propose text classification strategies that will perform efficiently even when additional information besides the pure text, such as the hierarchical structure of the texts or date of publication, etc. is not available [5].

2.2 Sentimental Analysis of Twitter Data Using Classifier Algorithms

Authors Sharvil Shah, K Kumar, and Ra. K. Saravanaguru offer a method to detect a user's current attitude toward a specific issue in this work. They have provided not only a binary classification of positive and negative data, but also a hashtag classification for topic modeling, an emoticon analysis for assessing post polarity, and multiple language support using tools like Google Language Detector and Langid.

They've also used Google Chart Tools to create a graphical depiction of the sentiment analysis. The polarity shifter using Naïve Bayes classification algorithm and topic modeling are two critical steps in their method, which leads to a high 81% accuracy [6].

2.3 Event Classification and Retrieving User's Geographical Location Based on Live Tweets on Twitter and Prioritizing Them to Alert the Concern Authority

Authors Sarthak Vage, Sarvesh Wanode, Kunal Sorte, and Dipak Gaikar have presented a system where the user can enter keywords related to a situation and the application runs live to show the priority and classified tweets dynamically. The proposed method employs the Naive Bayes algorithm, a stochastic model that belongs to a class of simple probabilistic classifiers based on Bayes theorem application.

Various models were tested to classify as per priority, with XGBoost, a decision tree-based Machine Learning method that uses a gradient boosting framework, providing the best results. The geo-location of the tweet is an essential characteristic as it's necessary to know the user's location in specific scenarios [7].

2.4 Automatic Classification of Disaster-Related Tweets

In this work, authors Beverly Estephany Parilla-Ferrer, Proceso L. Fernandez Jr., and Jaime T. Ballena IV have constructed a machine learning model that categorizes disaster-related tweets as informative or uninformative, as well as assesses the performance of two of the most used machine classification methods, Naive Bayes and Support Vector Machine. Their findings show that SVM surpassed Naive Bayes in terms of accuracy, recall, AUC, and F-measure using 10-fold cross-validation, but Naive Bayes outperformed SVM in precision [8].

2.5 Comparing BERT Against Traditional Machine Learning Text Classification

In this work, four distinct text classification experiments have been carried out by the authors Santiago Gonzalez-Carvajal and Eduardo C. Garrido-Merch´an. In order to compare the results, they have employed two distinct classifiers in all of the experiments: the BERT classifier and a conventional classifier that trains machine learning algorithms in features retrieved by the Term Frequency - Inverse Document Frequency (TF-IDF) technique. Their findings show that BERT not only outperforms the traditional NLP approach but is also comparatively easier to implement [9].

2.6 Usage and Analysis of Twitter During 2015 Chennai Flood Towards Disaster Management

This work by authors Meera R. Nair, G. R. Ramya, and P. Bagavathi Sivakumar focuses on tweets regarding the 2015 Chennai Flood. Extensive data exploration and analysis of these tweets show that the tweets fall under five distinct categories. Three machine learning techniques were employed to classify the tweets: Random Forests, Decision Trees, and Naive Bayes. Precision, recall, and F-measure was used to evaluate performance. It was found that Random Forests was more suited for twitter analysis and classification. This paper also focuses on identifying the most influential users of the Chennai flood using data mining techniques [10].

2.7 A Comparative Analysis of Machine Learning Techniques for Disaster-Related Tweet Classification

In this work, authors Abhinav Kumar, Jyoti Prakash Singh, and Sunil Saumya have implemented five deep neural network-based models and seven machine learning classifiers on tweets related to earthquakes, hurricanes, floods, and wildfires. The tweets are classified into six different categories. Their findings show that deep neural networks give better classification results than traditional classifiers even when there is data imbalance. GloVe embeddings gave the best results for wildfire-related tweets while Crisis embeddings proved to be better in case of earthquakes [11].

2.8 Multimodal Analysis of Disaster Tweets

In this work, authors Akash Kumar Gautam, Luv Misra, Ajit Kumar, Kush Misra, Shashwat Aggarwal, Rajiv Ratn Shah have analyzed multimodal data related to various natural calamities from the CrisisMDD dataset consisting of both textual and image data extracted from Twitter. They have proposed a novel decision diffusion technique in this paper to classify the data as informative or non-informative [12].

3 Methodology

The basic outline of the project is as follows (Fig. 1):

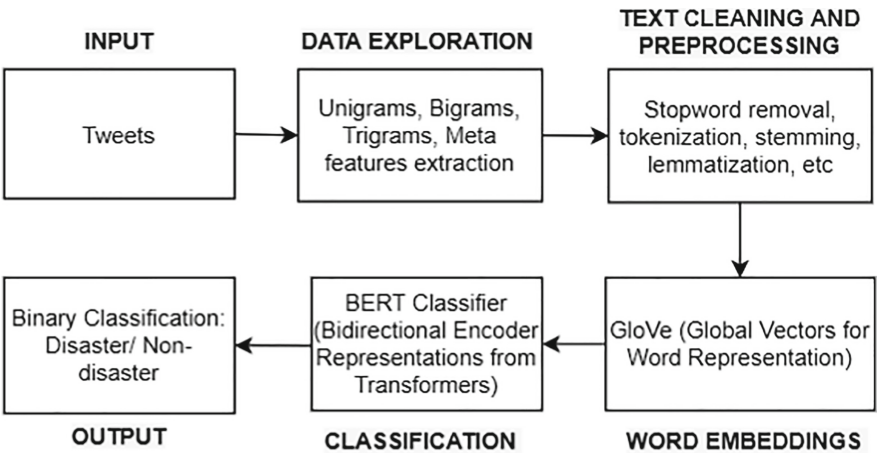

Fig. 1. Flowchart

3.1 Data Exploration

The dataset we used contains 7613 rows of text along with the location, keywords and the target, that is, whether it is a disaster or not. For testing purposes, we have a separate CSV containing 3263 rows. A small dataset gives rise to the problem of

overfitting especially in a dataset involving textual features. Thus this amount is sufficient to create a robust and reliable model. Data gathered from tweets contains a lot of ambiguity and hence needs to be processed before passing it to the model. For this firstly we segregate the data into unigrams, bigrams, and trigrams, as usually unigrams are sufficient for training but using bigrams and trigrams allows for complex meanings to be processed by the model efficiently (Fig. 2).

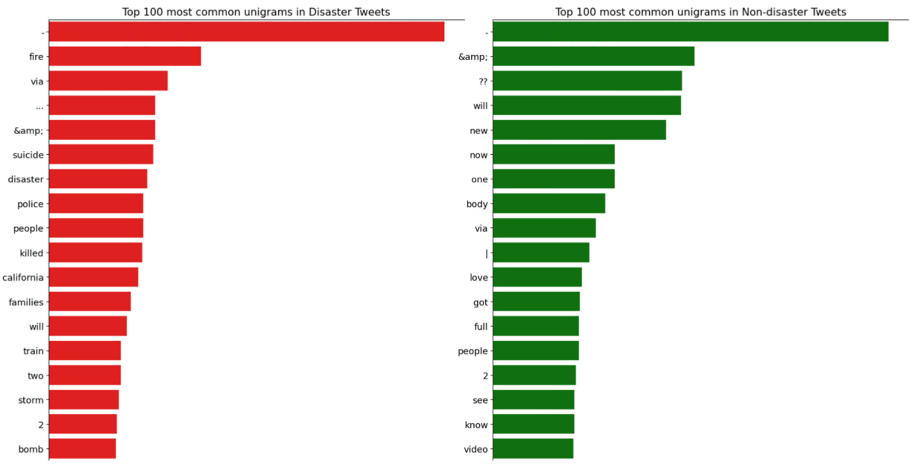

Fig. 2. Unigrams list

The most frequent unigrams in disaster tweets already provide information about disasters. Some of the terms are quite difficult to use in other situations. Verbs are the most prevalent unigrams in non-disaster tweets. Because the phrases are originating from individual users, the majority of them have an informal active structure (Fig. 3).

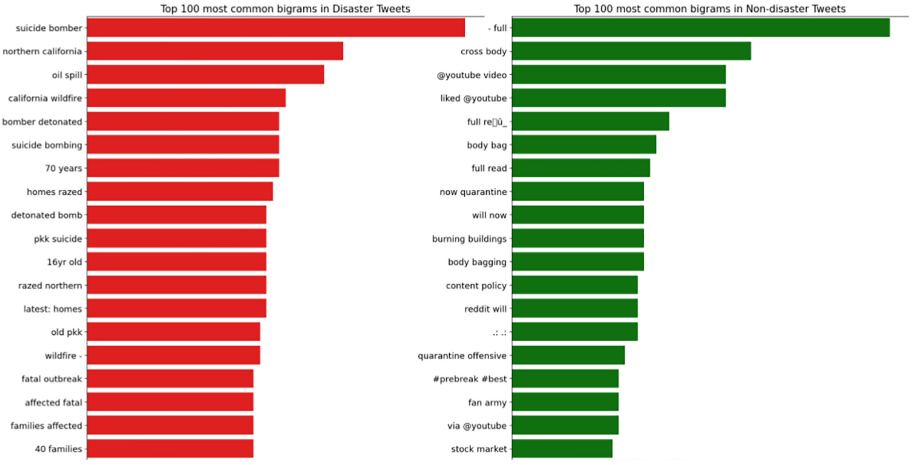

Fig. 3. Bigrams list

The most common bigrams provide additional information about the disasters as compared to unigrams. However, punctuations have to be removed from these words (Fig. 4).

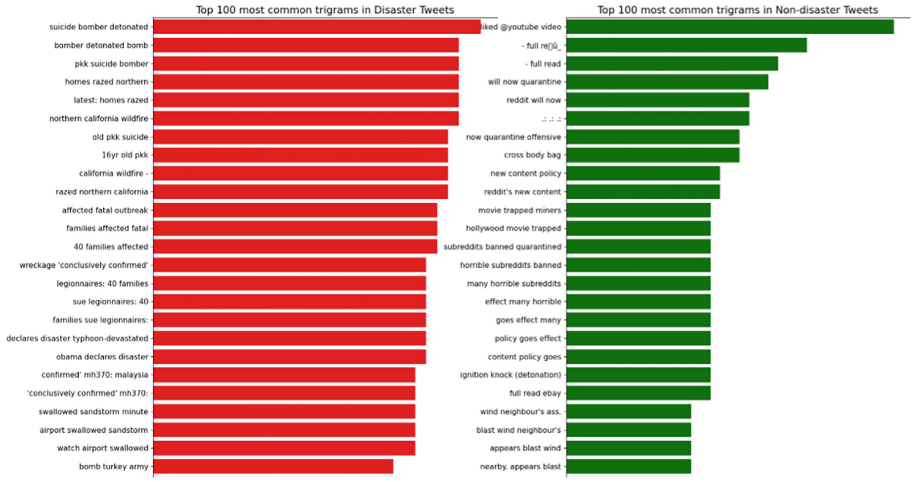

Fig. 4. Trigrams list

Trigrams are extremely similar to the most prevalent bigrams in disaster tweets. They provide a lot of information related to disasters, however, they may not supply any extra information beyond bigrams.

3.2 Meta Features Extraction

Meta-feature distributions in classes and datasets can assist identify disaster tweets. Because the majority of the tweets come from news organizations, it may be deduced that disaster tweets are written more professionally with lengthier words than non-disaster tweets. Because they come from individual people, non-disaster tweets have more mistakes than disaster tweets. The meta-features used for the analysis are:

- word_count number of words in the text
- unique_word_count number of unique words in the text
- stop_word_count number of stop words in the text
- url_count number of URLs in text
- mean_word_length average character count in words
- char_count number of characters in the text
- punctuation_count number of punctuations in text
- hashtag_count number of hashtags (#) in text
- mention_count number of mentions (@) in text

All of the meta-features in the training and test sets have extremely similar distributions, indicating that the training and test sets are from the same sample. All of the

meta-features provide information on the target, although some of them, such as URL count, hashtag count, and mention count, are useless. For disaster and non-disaster tweets, however, word count, unique word count, stop word count, mean word length, char count, and punctuation count have significantly different distributions. These characteristics are beneficial in models.

3.3 Text Cleaning and Preprocessing

The preprocessing of tweets is a crucial step for any text mining task. Tweets often include personal opinions and views in addition to factual information. Tweets that have not been preprocessed are very unstructured and may contain a lot of redundant data. Text cleaning helps us to get rid of noisy and inconsistent data which may hamper the efficiency of our machine learning model. Multiple steps are taken in the preprocessing of tweets as mentioned below:

- Lowercasing the entire text
- Removal of punctuation marks
- Removal of URLs
- Removal of hashtags and usernames
- Removal of numbers and special characters
- Correction of typos, informal abbreviations are written in their long forms.
- Tokenization
- Stemming and Lemmatization
- Removal of stop words

Tokenization is the process of converting a sequential piece of text into smaller units called tokens. These tokens are then used to build a vocabulary. In traditional NLP approaches, the vocabulary is used as a feature to train the model.

Both Stemming and lemmatization are methods to convert words into their root forms. This eliminates the need to store all forms of words and prevents the overfitting of the model.

Stopwords are common words such as ‘the’, ‘on’, ‘an’, etc. which do not provide any important meaning to the text. It is important to remove such stopwords from the data as they do not contribute to the classification task and take up the valuable processing time.

3.4 Cardinality and Target Distribution

Locations and keywords are a part of the tweets. Locations are generally given by the user. Hence, there are far too many unique values for the locations. Therefore, locations should not be used as a feature. Keywords, on the other hand, are context-specific. So a keyword can be used as a feature. It is also feasible to utilise target encoding on keywords if the training and test sets are from the same sample (Figs. 5 and 6).

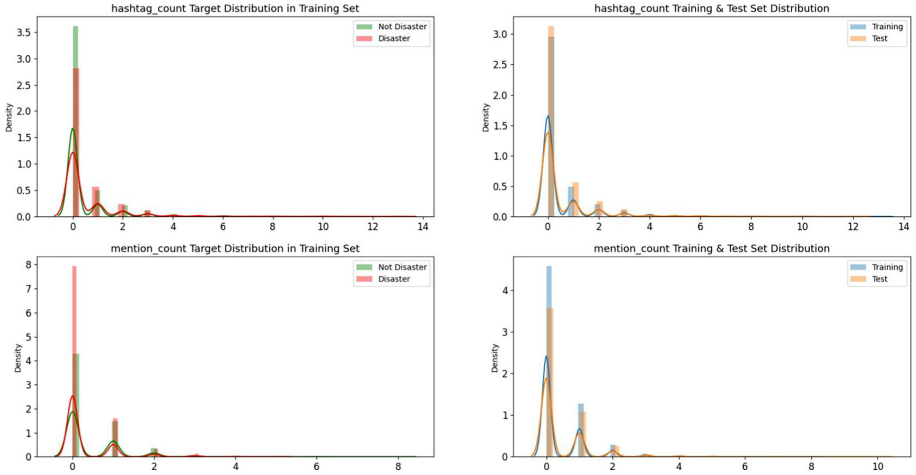

Fig. 5. Hashtag and mention count target distribution

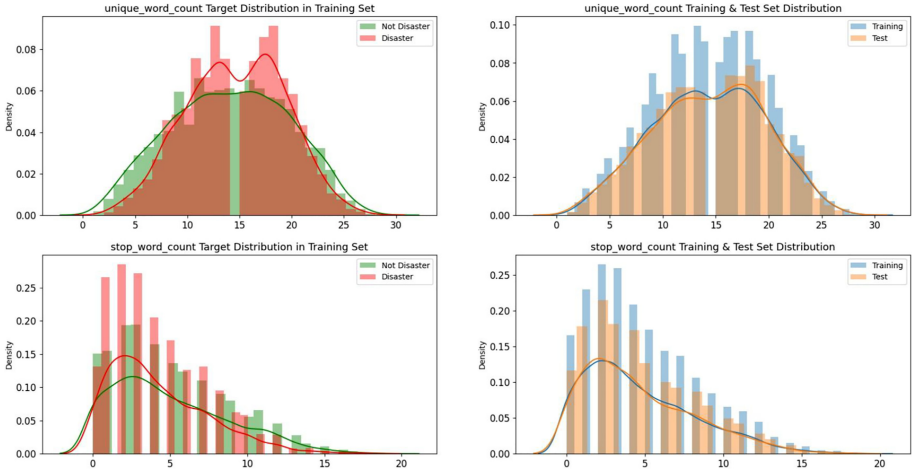

Fig. 6. Unique word count and stop-word count target distribution

3.5 Target Features

The class distributions for 0 (Not Disaster) and 1 (Disaster) are 57% and 43%, respectively. Because the classes are almost similar in size, cross-validation does not need stratification by the target (Fig. 7).

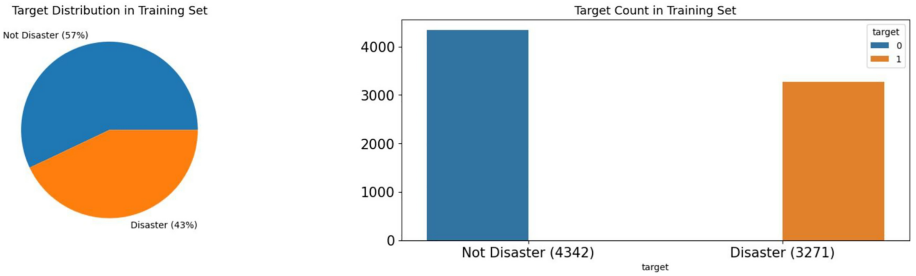

Fig. 7. Target distribution graph

3.6 Architecture of the Disaster Detector Function

The preprocessed input text is fed into the DisasterDetector(), a wrapper function, that includes the above-mentioned cross-validation and metrics. The FullTokenizer class [15] performs the tokenization of input text. The max sequence length (set to 128) property can be used to adjust the length of text sequences. During the learning phase, parameters like Learning Rate (0.0001), Epochs (10), and Batch size (32) produced the best results. After the last layer of BERT, no dense or pooling layers are introduced. Because other optimizers have issues converging, SGD is employed as an optimizer (Fig. 8).

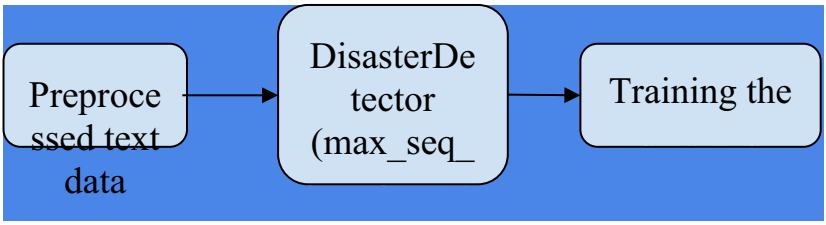

Fig. 8. Architecture

3.7 Word Embedding Using GloVe

Word embedding is an essential part of any natural language processing problem. Machine learning models cannot directly work on textual data. Hence, it is necessary to convert the text into a numerical format. Word embeddings are numerical representations of text. There are several methods for this such as Bag of Words.

TF-IDF, Word2Vec, etc. In this work, we have employed Stanford's GloVe (global vectors for word representation) Pre-Trained Word Embedding [13]. The gloVe helps to capture the semantic and syntactic meaning of words by deriving the relationship between the words using simple statistics. GloVe encodes the information of the probability ratio in the form of word vectors. Using a pre-trained model has two major advantages: Firstly, as it is trained on very large datasets having a rich vocabulary, it

performs well even if the data has a lot of rare, uncommon words. Moreover, they are much faster than learning embeddings from scratch. Another benefit of GloVe is that, unlike Word2vec, it includes global statistics (word co-occurrence) to generate word vectors instead of just depending on local statistics (local context information of words) as it uses matrix factorization techniques on the word-context matrix.

3.8 Classification Using BERT Model

This research requires the BERT family of transformers from the TensorFlow Models repository on GitHub [14]. Each token of input text is processed by the BERT family of models (utilizing a Transformer encoder architecture), in the full context of all tokens before and after, therefore the name: Bidirectional Encoder Representations from Transformers. It uses L = 12 hidden layers, that is, Transformer blocks, a hidden size of H = 768, and A = 12 attention heads. The entire BERT is pre-trained for English on Wikipedia and BooksCorpus. All the text is converted to lower-case before tokenization into word pieces, and all the accent markers are cleaned. This Neural network-based approach is implemented as traditional techniques like Naive Bayes, Term Frequency Inverse Document Frequency (TF-IDF), Count Vectorizer only take into account the frequency of the words and not their semantics, that is, their meaning, hence the performance hinders when the amount of data and the complexity of the sentences increases. Therefore, implementing a Neural Network centric method (BERT) is more preferential.

4 Results

After data preprocessing and feature extraction, the dataset was trained on the BERT layer. We froze the outcomes of training and testing at 10 epochs as the curve started plateauing after 6 epochs, hence it was redundant to train the model for more epochs. The classification reports presented in the figure clearly show the values of different parameters used for evaluation after training the model for 2 folds segregated using Stratified K-Fold. The results obtained are Training Precision: 0.873517, Training Recall: 0.854729, Training F1: 0.860643, Validation Precision: 0.824855, Validation Recall: 0.809625, Validation F1: 0.814241 for 0th Fold and Training Precision: 0.884135, Training Recall: 0.867071, Training F1: 0.872638, Validation Precision: 0.85964, Validation Recall: 0.842464, Validation F1: 0.847904 for 1st Fold. From these results, we can infer that the BERT model is successful in understanding the texts and can be employed in detecting disasters from tweets (Fig. 9).

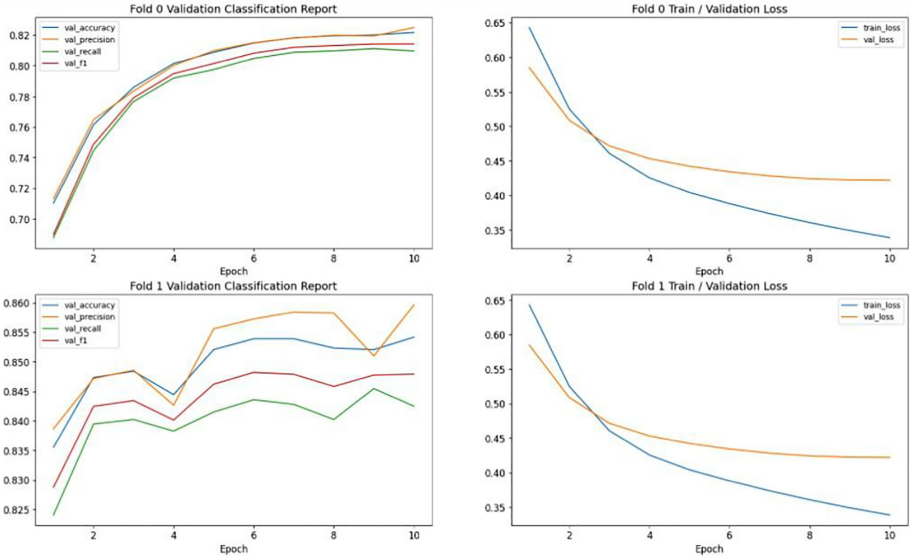

Fig. 9. Classification report

5 Conclusion

Twitter has proved to be a rich source of information for data mining and analysis. This can be especially useful during calamities and disasters as many individuals, NGOs, and government agencies rely on Twitter for information transfer. In this work, we have dealt with the binary classification of disaster-related tweets. We have employed trained glove embeddings and BERT classification models for this purpose. Utilizing GloVe we generated word vector representations using aggregated global word-word co-occurrence statistics. The actual classification task is done using Google's BERT model. BERT scans the full sequence of words at once, unlike typical NLP models that read the text from left to right or right to left. BERT utilizes a Transformer, which is essentially a mechanism for establishing relationships between the words in a dataset. The predictions made are robust and can accurately differentiate between a disaster-related tweet (target) and an unrelated tweet. In the future, we can make it a multi-class classification problem, wherein we can also predict what kind of disaster is mentioned in the tweet. e.g. Earthquake, Forest Fire, Tsunami, Flood. A web application that informs the concerned government authorities in case of disasters through email and SMS notifications to send relief is a plus.

References

1. https://www.omnicoreagency.com/twitter-statistics/
2. https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets
3. https://www.smrfoundation.org/nodexl/
4. https://www.kaggle.com/c/nlp-getting-started/data
5. Ikonomakis, E., Kotsiantis, S., Tampakas, V.: Text classification using machine learning techniques. WSEAS Trans. Comput. **4**, 966–974 (2005)
6. Shah, S., Kumar, K., Sarvananguru, R.K.: Sentimental analysis of Twitter data using classifier algorithms. Int. J. Electr. Comput. Eng. **6**(1), 357 (2016). https://doi.org/10.11591/ijece.v6i1.pp357-366
7. Vage, S., Wanode, S., Sorte, K., Gaikar, D.: Event Classification and Retrieving User's Geographical Location based on Live Tweets on Twitter and Prioritizing them to Alert the Concern Authority (2020)
8. Parilla-Ferrer, B.E., Fernandez, P.L., Ballena, J.T.: Automatic classification of disaster-related tweets. In: Proceedings of the International conference on Innovative Engineering Technologies (ICIET), December 2014, vol. 62 (2014)
9. González-Carvajal, S., Garrido-Merchán, E.C.: Comparing BERT against traditional machine learning text classification (2020). https://arxiv.org/abs/2005.13012
10. Nair, M.R., Ramya, G.R., Sivakumar, P.B.: Usage and analysis of Twitter during 2015 Chennai flood towards disaster management. Procedia Comput. Sci. **115**, 350–358 (2017)
11. Kumar, A., Singh, J.P., Saumya, S.: A comparative analysis of machine learning techniques for disaster-related tweet classification. In: 2019 IEEE R10 Humanitarian Technology Conference (R10-HTC) (47129), pp. 222–227 (2019). https://doi.org/10.1109/R10-HTC47129.2019.9042443
12. Gautam, A.K., Misra, L., Kumar, A., Misra, K., Aggarwal, S., Shah, R.R.: Multimodal analysis of disaster tweets. IEEE Fifth Int. Conf. Multim. Big Data **2019**, 94–103 (2019). https://doi.org/10.1109/BigMM.2019.00-38
13. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. EMNLP. **14**, 1532–1543 (2014). https://doi.org/10.3115/v1/D14-1162
14. https://github.com/tensorflow/models/tree/master/official/nlp/bert
15. https://github.com/tensorflow/models/blob/master/official/nlp/bert/tokenization.py