CBS
COPENHAGEN
BUSINESS SCHOOL
HANDELSHØJSKOLEN

Identifying Tweets Related to Emergencies using Natural Language Processing and Deep Neural Networks

A study of model efficiency for unstructured text data

Ramon Daniel Habtezghi (S149855)
Marten Jakob Stubenrauch (S149896)
Marin Elisabeth Henrica Mes (S149899)
Oskar Munck af Rosenschöld (S149873)

Department of Digitalisation
MSc Business Administration and Data Science
Natural Language Processing and Text Analytics - KAN-CDSCO1002U

N.o. pages: 8
N.o. characters: 29 857 (14 standard pages)
Date of submission: 28th May 2022

Abstract

With the world becoming ever more digitalized and connected, it is now possible for anyone to post and broadcast any information to society, especially through platforms like Twitter. These developments have led to new requirements regarding screening and verifiability of the information on these platforms. In this paper, we research to what extent certain Natural Language Processing techniques can aid governmental institutions and companies alike in their efforts to take advantage of the data generating machine Twitter. We test the performance of four different machine learning models in classifying tweets into one of two categories: emergency tweets or non-emergency tweets. We compare a Naïve Bayes Classifier, a Support Vector Machine, a Recurrent Neural Network, and a pretrained BERT model, each of which were trained on 7,613 tweets. Preprocessing steps such as tokenization, stop words removal, and lemmatization were employed in order to make the input fit for use. We find that the Support Vector Machine is the best classifier for the used dataset based on the evaluation metrics recall and F1-measure, and since it is one of the fastest models out of the four.

Key words: binary classification, topic modelling, emergency tweets, Naïve Bayes Classifier, Support Vector Machine, Recurrent Neural Network, BERT.

1. Introduction

In an ever-more connected world, Twitter has become one of the quickest ways to communicate. Via instant text posts anyone can post and relay any information that is broadcasted into society. These developments have led to new requirements regarding screening and verifiability of the information floating around the platform. Subsequently, use cases for leveraging Natural Language Processing (NLP) and other Machine Learning (ML) techniques to process and monitor this unstructured text data has increased.

In this paper we gauge how NLP techniques can aid in the classifying information (tweets) in a domain (Twitter) characterized by a high level of unstructured and non-standardized data. We explore how the contents of a tweet can determine if it is related to an emergency or not. We compare a Naïve Bayes Classifier (NBC), a Support Vector Machine (SVM), a Recurrent Neural Network (RNN), and the pretrained BERT model in their performance of classifying tweets into one of two categories: emergency tweet or non-emergency tweet. The results can support governmental institutions and companies alike in their efforts to take advantage of the data generating machine Twitter, as the models can be used as a vessel to support in the fighting of emergencies such as virus outbreaks, (wild)fires, or other (non) natural disasters.

To perform said analyses, a dataset by Appen was utilized, containing over 10,000 tweets labelled as a real emergency or not (data.world, 2016). The steps taken before, during and after are described in this paper. It starts off by describing the exploratory data analysis process and any preprocessing steps that were taken, including topic modelling using Latent Dirichlet Analysis and Top2Vec. Following this the chosen classifiers are described, their results are presented and compared, and errors are analyzed. This paper concludes by acknowledging its limitations and providing suggestions for future work and usability.

2. Related Work

This paper will aim to achieve binary classification of disaster tweets by using popular natural language processing models, building upon previous literature. Research by Sit, Koylu, and Demir (2019) concerns identifying whether a dataset of tweets relating to hurricane Irma were related to the hurricane, based on their spatial, temporal and semantic components. Thereafter they employed Latent Dirichlet Allocation (LDA) to discover the complex categories of the tweets related to the hurricane. The authors evaluated an LSTM Neural Network, a Convolutional Neural Network, a Logistic Regression, a Gaussian Support Vector Machine, a Linear Support Vector Machine, and a Ridge Classifier for the task of classifying the tweets. The results showed that the LSTM neural network yielded the best results evaluated by the F1 score, yielding 75.14 percent. Through the employment of LDA, the authors found twelve dis-

tinct categories in the data. Temporal analysis of the categories was performed showing that the categories alternated in prevalence depending on the stage of the emergency.

Muaad et. al (2021) compared a Support Vector Machine, a Random Forest Classifier, a Logistic Regression, a Passive Aggressive Classifier, a Decision Tree Classifier, a KNN classifier and the Arabic implementation of BERT to find misogynistic content on Twitter written in Arabic. They evaluated the models both on their ability to do binary classification as well as multi classification and found for the binary classification that SVM and BERT achieved the best F1 scores which they primarily used as evaluation metric.

Riyadh, Alvi and Talukder (2017) used a Multinomial Naïve Bayes classifier to do emotional analysis of tweets, classified into five emotions. They concluded that the Naïve Bayes Classifier performs well with language data, but that Part of Speech tagging did not increase the accuracy of the results.

Risch et al. (2019) performs binary language classification of German tweets as offensive and non-offensive using the BERT neural network pretrained on 12 GBs of the German language. They conclude that the model achieves a macro-average F1 score of 76.4 percent and that increasing training epochs from one to four yielded limited enhancements.

Given the findings related to model selection for binary tweet classification from the related literature, we have chosen to evaluate the performance of a Naïve Bayes classifier, a Bi-directional Long-Short term memory RNN, a Support Vector Machine, and the Google BERT neural network for classifying emergencies. Next to classifying the tweets, we employ two topic modelling techniques during the exploratory data analysis to find the most important topics of this dataset and get an overview of what the tweets are about.

3. Methodology

3.1. Dataset Description

The dataset used in this paper has been curated by Appen, a global organization that provides data sourcing, data annotation, and model evaluation, and have made the dataset available via a Kaggle competition (Appen, 2022). The dataset holds 10,876 tweets and has been hand-classified to denote whether a tweet is related to a real emergency or not. The original dataset contains 13 features (figure 2). In terms of data quality, five out of 13 features contain null values.

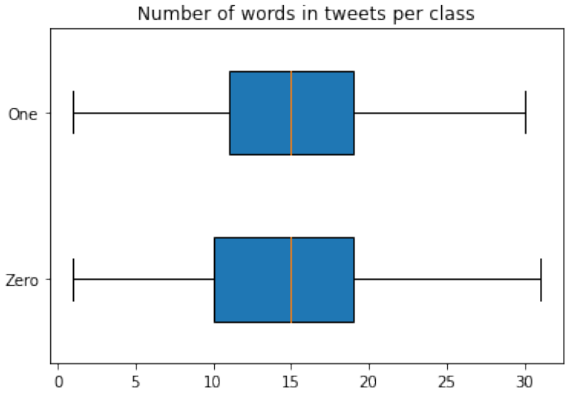


Figure 1: Boxplot showing average length per tweet per class. 'One' corresponds to tweets related to emergencies and 'Zero' those unrelated.

3.2. Exploratory Data Analysis

3.2.1. General

Within the dataset, a slight class imbalance is observed as the tweets classified as emergencies and the tweets classified as non-emergencies make up 43 percent and 57 percent respectively. The total dataset contains 162,293 words, the longest tweet is 31 words, the shortest tweet is one word, and the average tweet length is around 15 words for both classes (figure 1). The most common word after preprocessing is 'fire' which appears 408 times and the most common special character is '/' appearing 20,919 times.

3.2.2. Topic modelling

To understand our dataset better, we used topic modelling to create an overview of what the real emergency tweets were expressing. Topic modelling is an unsupervised machine learning technique that tries to extract topics by clustering together similar words within a dataset. Once the topics have been extracted, it is up to humans to interpret the found topics. We applied two topic modelling techniques: Latent Direchlet Allocation (LDA) and Top2Vec. The main difference between the two is that LDA assumes that the number of topics is known, whereas for Top2Vec the number of topics is computed within the model.

3.2.3. Preprocessing

Before employing the two topic modelling models, the dataset needed to be prepared. Since we aim to find the topics only of the emergency tweets, the non-emergency tweets were filtered out. Following this, the following preprocessing techniques were performed: (1) the removal of numeric strings, URLs, punctuation, short words, and capital letters, (2) the expansion of contractions, (3) the tokenization

Features	Type	Description	Null values
_unit_id	Float	-	0
_golden	Bool	-	0
_unit_state	Str	-	0
_trusted_judgments	Float	The number of trusted judgements	0
_last_judgment_at	Date	The data of the last judgement.	8543
choose_one	Str	this denotes whether a tweet is about a real emergency or not	0
choose_one:confidence	Float	The confidence of the manual classification	0
choose_one_gold	Str	-	10789
keyword	Str	A particular keyword from the tweet	87
location	Str	The location the tweet was sent from	3638
text	Str	The text of the tweet	0
tweetid	Float	A unique identifier for each tweet	0
userid	Str	A unique identifier for each user	87

Figure 2: Overview of the dataset

of tweets and (4) the lemmatization of tokens. This leaves us with a dataset of 9655 unique tokens.

3.2.4. Latent Direchlet Allocation (LDA)

An LDA model requires input in the form of a document-term matrix. This was created with aid of a dictionary assigning a unique ID to each word, and with the bag of words approach. With the input ready, the number of topics that the dataset should be divided into needed to be specified. Previous research has shown that the best way to determine this is by evaluating the model's coherence score for different number of topics, where a higher coherence score implies better human interpretability (Röder, Both & Hinneburg, 2015). The coherence score is a score based on the degree of semantic similarity between high scoring words within a topic (Kapadia, 2019). After plotting the coherence score for different number of topics, we find that the best number of topics is eight (Appendix A).

The results of the LDA model depict the eight different topics defined by a combination of keywords with attached weights (Appendix B). It appears that the LDA model finds topics that range relatively widely in their keywords. For instance, within one topic words such as 'wildfire', 'nuclear' and 'weather', or 'train', 'flood' and 'refugee', are found, and the word 'fire' appears in 4 out of 8 topics. Nevertheless, the 8 topics can be interpreted as shown in figure 3, offering a general overview on the type of disasters tweeted about in the emergency tweets.

Topic 0:	California wildfires disaster
Topic 1:	Nuclear disaster
Topic 2:	Crash disaster
Topic 3:	Burning building disaster
Topic 4:	Train disaster
Topic 5:	Oil disaster
Topic 6:	Family disaster
Topic 7:	Suicide bombing disaster

Figure 3: The interpreted topics after LDA

3.2.5. Top2Vec

The second topic modelling model that was employed is the Top2Vec model. In contrast to LDA, Top2Vec requires a list of strings as input and does not assume that the number of topics is known in advance. Top2Vec uses embedded vectors, dimensionality reduction and clustering to derive the topics in a dataset. As a clustering technique, HDBSCAN is used since this technique does not force each data point in a cluster, but also considers outliers.

The Top2Vec model finds 42 topics across the dataset and like the LDA model, its results depict the different topics defined by a combination of keywords with attached weights (Appendix C). Top2Vec finds 42 topics across the dataset, with the substantial one being that of the outliers, containing more than 1800 datapoints. The next most substantial topics contain 505, 277, 229, 144 and 135 data points respectively. Since the number of topics is almost six times larger than in the LDA model, the topics resulting from the Top2Vec are more specific and easier to interpret. The 8 biggest topics can be interpreted as shown in figure 4.

It can be said that even though the two topic modelling techniques work quite differently, they end up finding similar topics, thus providing us with similar summaries of the data. Having a broad sense of what the emergency tweets in the dataset are about aids us in understanding what kind of data the dataset contains and can help us in understanding future results.

Topic 0:	Death
Topic 1:	Nuclear disaster
Topic 2:	Terrorism disaster
Topic 3:	California wildfires
Topic 4:	Burning buildings disaster
Topic 5:	Online disaster
Topic 6:	Suicide bombing disaster
Topic 7:	Train derailment disaster

Figure 4: The interpreted topics after Top2Vec

3.3. Data preparation

Figure 5 describes the preprocessing steps that were taken for the data to be suitable input for the classification models. For optimization of the models, we used a training, validation, and test split of 70, 15, and 15 percent. After optimization, we used a train and test set of 70 and 30 percent. All preprocessing steps will be described in detail in the following sections.

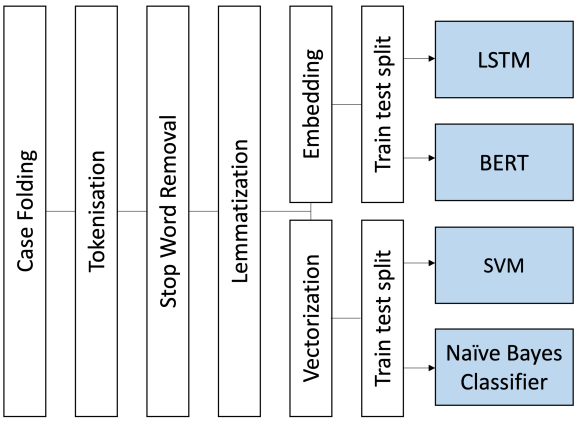


Figure 5: Preprocessing steps

3.3.1. Dataset adjustment

From the initial dataset, we considered the features _unit_id, _golden, _unit_state, _trusted_judgements, _last_judgement_at, choose_one: confidence and choose_one_gold to be irrelevant for our research. Therefore, we decided to drop these irrelevant fields.

An additional step in adjusting the dataset was the binary encoding of the choose_one column which classifies the tweets as relevant or not relevant. We created a new column target for this binary encoding where a one denotes 'relevant' and a zero denotes 'non-relevant'. This way the dataset was reduced to having seven features.

The location feature has been disregarded in this project due to 30 percent null values. The portion of null values in the keyword and userid column cover less than 10 percent of the entire column. Fortunately, the primary features, text (the actual tweet) and target (whether it is a real emergency or not), do not contain any null values and therefore no further action with regards to the null values was required.

3.3.2. Case folding, tokenization, stop word removal and lemmatization

The first step in the data preprocessing was case folding, as this is a prerequisite for most further preprocessing tasks. Following the case folding, the tweets were tokenized, i.e., converted into a list of strings. Tokenization is done in order to efficiently engage in further preprocessing steps. Next up, stop word removal was executed to reduce the dimensionality of the data. Stop words are words that in general do not add meaning to the sentence and can therefore create noise, making it harder for the model to classify the data. According to Saini and Rakholia (2016), any language contains around 200 stop words. In this paper, the removed stop words are based on the python dictionary NLTK English which contains 179 words

(Bird, Loper & Klein, 2009). Examples of stop words that were removed are 'there', 'when' and 'me'.

Once the stop words were removed, the remaining words were lemmatized. Lemmatization is the morphological processing of language where all words are brought to their root (Jurafsky & Martin, 2021). For example, 'run', 'ran' and 'running' are all reduced to their root 'run' after being lemmatized. This way, the complexity of the data is further reduced, making it easier for the models to process. Examining the effect of the lemmatization on our dataset, it is predominantly plural forms of words that have been reduced to their singular form.

After applying said preprocessing steps, the dataset dimensionality was successfully decreased by 1.69 percent (- 618 unique words) and the total dataset was reduced by 25.01 percent from 162 293 to 121 709 words (- 40 584 words).

3.3.3. Vectorization and Embedding

For two of the four models, the NBC and SVM model, the data needed to be vectorized for it to be usable input. We utilized the bag of words approach to do this, which entails transforming each instance to a numerical vector with token counts enumerated by the vocabulary order. Since the vocabulary order is random, the information about the positioning of the words in each instance is lost. This information loss is however necessary since NBCs and SVMs are unable to handle it.

For the other two models, the LSTM and BERT models, the positional information is kept and words are embedded instead of vectorized. Since both models utilise different embedding strategies, how the embedding was executed will be handeled in section 3.4.3 and 3.4.4 for LSTM and BERT respectively.

3.4. Modelling Framework

This section will explain the architecture behind the four models that were analyzed in this paper.

3.4.1. Naïve Bayes Classifier (NBC)

The NBC is based on Bayes rule, which relies on a simple representation of a document as a bag of words (Vikramkumar & Trilochan, 2014). This bag of words representation contains all words across the dataset and assumes that the position of each word does not matter. Under the assumption that the different probabilities of each word are independent of the occurrence of other words and of the class it is in, the algorithm classifies the tweet into the class with the highest likelihood given the words in the tweet.

When predicting the different classes we used the fractional word counts tf-idf to improve the models performance. Additionally, we added LaPlace smoothing to take into account that our test set could entail unknown words. NBC is shown to be good for domains with many equally important features. This could pose a challenge in emergency tweet detection, as sometimes certain words are more important than others. However, we will use the NBC as a baseline for comparisons as it provides a dependable text classification. Additionally, the NBC is inherently fast, fitting to the domain of emergency tweets as decisions to act upon these tweets must be made within split seconds.

3.4.2. Support Vector Machine (SVM)

SVM's are used for a multitude of use cases within machine learning and can be applied to tasks such as linear or non-linear classification, regression, and outlier detection. Additionally, they have great applicability for classification tasks on small-to-medium sized datasets (Géron, 2019). The model determines decision boundaries (hyperplanes) to separate and classify data points. In the determination of the hyperplane, SVMs finds the closest data points from each class (support vectors) and establishes the margin, the distance between the support vectors, and the hyperplane. The model then optimizes for the largest margin possible. Like NBC, the Linear SVM in this paper utilizes the bag of words approach and TF-IDF to establish the frequency and importance of the words.

3.4.3. Recurrent Neural Network (RNN) - Long Short Term Memory Network (LSTM)

RNNs are a strong branch of deep learning methods for text classification as they take into consideration the dependencies between elements in a sequence. In our paper we employ a special kind of RNN capable of learning long-term dependencies: a LSTM (Hochreiter & Schmidhuber, 1997). LSTM models in general achieve high accuracy since they utilize three unique gates. When feeding in the tweets, the forget gate is responsible for the memory keeping function of LSTM networks, and thus responsible for keeping track of the interdependencies. The input gate selects which information gets fed into each of the other cells. Like the forget gate, the input gate uses a sigmoid function and has a binary output determining if information is passed through or not. The output gate follows a similar structure but determines which information is updated and remembered in the hidden state of the LSTM network. Before feeding our LSTM with the text we used an embedding technique to not only make the algorithm more efficient but also to help recognize similar meanings and interdependencies between words. An overview of

the LSTM node architecture can be found in Appendix D. To prevent our model from overfitting, we introduced a dropout function as this has been proven to minimize overfitting (Srivastava et. al., 2014). Binary Cross entropy will be used as the loss function. In total our LSTM has 394,000 parameters.

3.4.4. Bidirectional Encoder Representation Form Transformer (BERT)

BERT is a transformer-based pre-trained machine learning model developed by Google (Devlin et. al., 2019). Until its development, state-of-the-art NLP systems relied on gated RNNs, such as LSTM due to their memory ability and taking into consideration the interdependencies of words. BERT is achieving this by keeping the attention, but without recurrent sequential processing, thus minimizing training time. The architecture of the original BERT that we use in this paper utilizes two encoders where the first converts the input tweet into an embedding, and where the second supports the following decoder layer to focus on the relevant input tweet. The key layer to achieve this is the self-attention layer, which is used to detect and understand currently processed words. By refining these self-attention layers to be multi-headed self-attention layers, BERT focuses on distinct positions in the tweet and provides representation of subspaces focusing on these positions. This process enables the bidirectional processing of the input sequence. The multi-headed self-attention layer provides the decoder with the right sentence parts to focus on which will then be decoded and put into the final layer with a SoftMax activation function. The vectors created by the stack of decoders will be combined into a larger vector containing logits for each word in the class. The cell with highest probability will then be returned. Training is done by comparing the predicted class with the actual using the Cross Entropy loss. See Appendix E for an overview of the BERT architecture. In total the BERT model has 110 million parameters and is pre-trained on the BooksCorpus and English Wikipedia totaling 3.3 billion words.

3.5. Evaluation Metrics

To evaluate and compare the four models we use Accuracy, Precision, Recall, and the F1-measure. Accuracy describes the general ability of our models to classify the tweets in the correct class, emergency, or non-emergency. Precision encompasses how many of the tweets classified as an emergency, are an actual emergency, and looks to minimize false positives. Recall looks at the total of the true emergency tweets, how many the model managed to predict to be in the emergency class and looks to minimize the number of false negatives. Precision and recall often pose a tradeoff as one are stricter on false positives and the other on false negatives. Thus, we decided to also include the F1-measure as an evaluation metric, which is the average between recall and precision.

For our models we will particularly use Recall as our main evaluation metric. In the scenario of an emergency, it is important that the model detects it correctly, thus reducing false negatives. Given our use case, a decrease in precision and an increase in recall would be acceptable as we want to decrease false negative classification of emergencies which could lead to emergencies to be unattended. However, as we do not want to encourage resource waste by teaching our models to classify every tweet to be an emergency, we will also consider the F1-measure in our evaluation as it provides a balanced view on the two metrics precision and recall.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4. Results

4.1. Accuracy, precision, recall, and F1-measure

The test results show that each model has a strength where it outperforms the other models (table 1). The best performing model based on our two most important metrics, Recall and F1-measure, is the SVM model, with a score of 0.73 and 0.75 respectively. The NBC has a score of 0.65 for Recall and 0.72 for the F1-measure. The two neural networks LSTM and BERT follow with 0.69 and 0.68 respectively on Recall, and 0.69 and 0.72 respectively on the F1-measure.

Metric	NBC	SVM	LSTM	BERT
Accuracy	0,79	0,79	0,74	0,77
Precision	0,82	0,77	0,69	0,78
Recall	0,65	0,73	0,69	0,68
F1-measure	0,72	0,75	0,69	0,72

Table 1: Accuracy, Precision, Recall, F1-Measure for the four classifiers

The confusion matrices for each model show the predictions of each label against the actual label (appendix F). In our paper we use the confusion

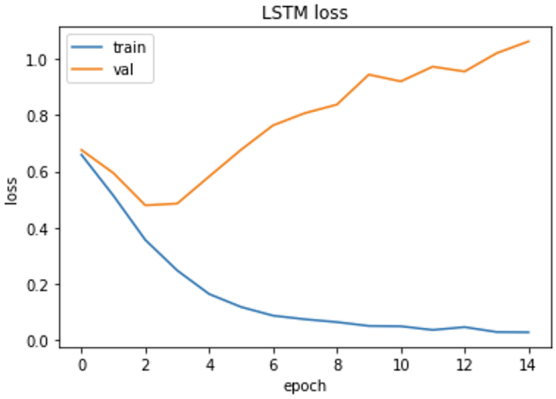


Figure 6: Train and validation loss of the LSTM model

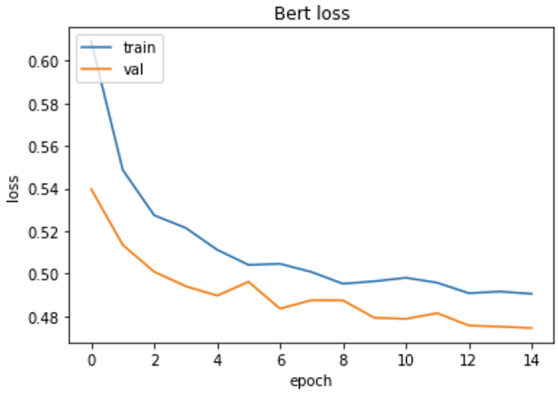


Figure 7: Train and validation loss of the BERT model

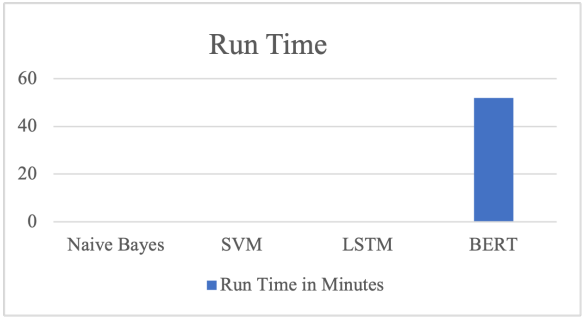


Figure 8: Run time of the four models

matrix to analyze how the false negatives of emergency tweets behave across the models as this is the most important metric for our use case. We find that all models, besides SVM, have an imbalance towards classifying a tweet to be non-emergency. This could be due to the slight imbalance between the two classes in the test set. The test set consists of 3,263 tweets in total, where 1,371 tweets are emergency tweets, and 1,892 tweets are non-emergency tweets. This slight class imbalance may be the reason we see a weaker performance across the models regarding recall, compared to precision.

For the neural networks LSTM and BERT, we present the training and validation loss in figures 6 and 7 respectively. Both models first trained on 15 epochs to optimize hyperparameters. LSTM shows significant overfitting from three epochs onwards. Due to this behavior our final model has run on three epochs to produce predictions. For BERT we see a slow convergence of validation and training model, additionally it nearly showed a complete convergence at epoch five. Thus, we decided to train our final BERT model over five epochs to produce predictions.

4.2. Run time analysis

When determining the quality of a model, it is also essential to consider the run-time of the models. Sometimes it is suggestable to use a less accurate model if it has a lower running time. The four models in this paper were run on a MacBook Pro with a 2 GHz Intel Core i5 and 16GB RAM. The run time of the four models are visualized in figure 8. The figure shows that BERT has the longest running time compared to the other models. In total BERT needed 180 minutes to run whereas the other three models needed less than a minute to run. By looking at the F1-measure, recall and run-time together, the SVM can be considered the best model for our use-case. The model shows high efficiency of running time combined with high evaluation metrics. It must be noted however that neural networks in general are highly dependent on the number of epochs when it comes to running time and might outperform as the size of the input data increases.

5. Discussion

5.1. Applicability of models

The results suggest that SVM outperforms the neural network models for this specific dataset. The SVM does not only show a high score on each of the evaluation metrics but is also quick in training and predicting.

We expected better results for the BERT model when training, as the BERT model bases its predictions on many parameters and previously learned relationships between embedded variables. However, it appears that the model overfits quite fast and is slow when training. BERT is extremely capable in translating tasks due to its large database, but classification seems to be one of its weaknesses. Our RNN model is the third strongest model based on speed and the chosen evaluation metrics. As a

next step, we suggest trying to improve the performance of the model by adding another layer of LSTM to the model and test if this makes the model perform better.

Lastly, the NBC, our baseline model for comparison, is the second to worst performing model based on our primary evaluation metrics. The NBC's inability to detect interdependencies between words and meaningful order within a sentence might explain this weakness. Additionally, it seems that the NBC is easily influenced by a small imbalance in classes due to its weight on the prior probability.

5.2. Error analysis

When analyzing the errors of the LSTM and SVM models we can spot two major groups where the errors might originate from. Firstly, for a human it would already be quite difficult to detect whether a tweet is an emergency tweet or whether it is merely commenting on something that is going on around the world. For example, the second tweet in table 2 is wrongly classified by the LSTM and SVM models as an emergency tweet. The tweet is tagging abcnews, and it seems that the author could be both informing the news channel, or simply commenting on some news story. This is hard for both computer and human.

Secondly, tweets could use certain unique language that is hard to understand for the algorithms. The third tweet in table 2 uses the word damages which could in this case be referring to an actual damage, or to a damage in another context, such as making damage in a video game. Similarly, the first tweet in table 2 uses past-tense which could lead the model into classifying it as a non-emergency tweet. Thus, looking at common misclassified tweets we can spot two possible reasons for misclassification: uncertainty whether informing or commenting, and unknown context.

"Bend Post Office roofers cut gas line prompt evacuation - http://t.co/6mF7eyZOAw"
"@abcnews A nuclear bomb is a terrible weapon!!"
"Got my first damage today [f*ckkkkkk]"

Table 2: Errors made by both LSTM & SVM

6. Conclusion & Future work

The goal of this paper was to develop a model that can support institutions in detecting tweets that can be linked to an emergency. Through this detection, institutions can take necessary steps to aid people in need or resolve an emergency. In today's world, where over 200 million tweets a day are posted, such a model would be extremely useful.

Our analysis suggests that the Support Vector Machine trained on our data set is the best model for prediction. Both based on performance in our primary evaluation metrics, recall and F1-measure, and run-time as it is one of the fastest models to produce a prediction. The deep neural networks show weaker performance but can be optimized by increasing the epochs in the case of BERT, or by adding more layers in the LSTM (RNN).

The main source of misclassification comes from the challenge to distinguish between information or commenting on emergencies, and the unknown contexts of tweets. These are general challenges of NLP and could be solved using more complex DNNs that use embeddings, indicating a great opportunity for future research.

Our study is limited using the information available. Twitter has a lot of unstructured text and links that can be used to refine models and predictions. Further connection between emergencies and a posted links, pictures or emojis should be made to find possible connections. Additionally, we did not use the location or keyword column in our dataset as we wanted to focus on the text alone. These two columns could entail more information about the tweets' content, context and connection to an emergency.

Future research should focus on optimizing the LSTM model to use embeddings to eliminate potential error sources. Additionally, the BERT model could be trained further as it shows potential to be improved by longer training runs. Additionally, an overall increase of the amount of tweets in the dataset could improve overall performance.

7. References

Appen (2022) Appen "About Us". Available at: https://appen.com/about-us/ [Accessed: 26 May 2022]

Bird, S., Loper, E. & Klein, E. (2009). Natural Language Processing with Python. O'Reilly Media Inc.

Crowdflower (2016) Disasters on Social Media dataset, Available online: https:// data.world/crowdflower/disasters-on-social-media [Accessed 28th April 2022]

Devlin, J., Chang, M., Lee, K. & Toutanova, K. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 2nd edition, O'Rielly: Sebastopol

Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation 1997; 9 (8): 1735–1780. DOI: https://doi.org/10.1162/neco.1997.9.8.1735 [Accessed 28 May 2022]

Jurafsky, D. & Martin, J. H. (2021) Speech and Language Processing, 3rd edition draft. [Unpublished manuscript], Available online: https://web.stanford.edu/ jurafsky/slp3/ [Accessed 25th May 2022]

Gundapu, S. & Mamidi, R. (2021). Transformer based automatic COVID-19 fake news detection system.

Le, X. H., Ho, H., Lee, G., & Jung, S. (2019). Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. Water. 11. 1387. 10.3390/w11071387.

Muaad, A. Y., Davanagere, H. J., Al-antari, M. A., Bibal Benifa, J.V., & Cola, C. (2021). AI-Based Misogyny Detection from Arabic Levantine Twitter Tweets, Computer Science and Mathematics Forum 1st International Electronic Conference on Algorithms, Available online: https://www.mdpi.com/2813-0324/2/1/15/htm [Accessed 28 May 2022]

Risch, J., Stoll, A., Ziegele, M., & Krestel, R. (2019). Offensive Language Identification using a German BERT model, Conference: Proceedings of the 15th Conference on Natural Language Processing, Available online: https://hpi.de/fileadmin/user_upload/fachgebiete/ naumann/publications/PDFs/2019_risch_hpidedis.pdf [Accessed 28 May 2022]

Riyadh, A. Z., Alvi, N., & Talukder, K. H. (2017). Exploring Human Emotion via Twitter, 20th International Conference on Computer and Information Technology, Available online: https://ieeexplore.ieee.org/abstract/document/8281813?casa_token=Hr5h1-hi-W0AAAAA:WyMk51oDNODq0j5OV3tXyoY91n_QPmBMO4Vaxt1a3xp1KMZvixx_vb8qqi8V7JeyQHy9nLZH4jlO [Accessed 28 May 2022]

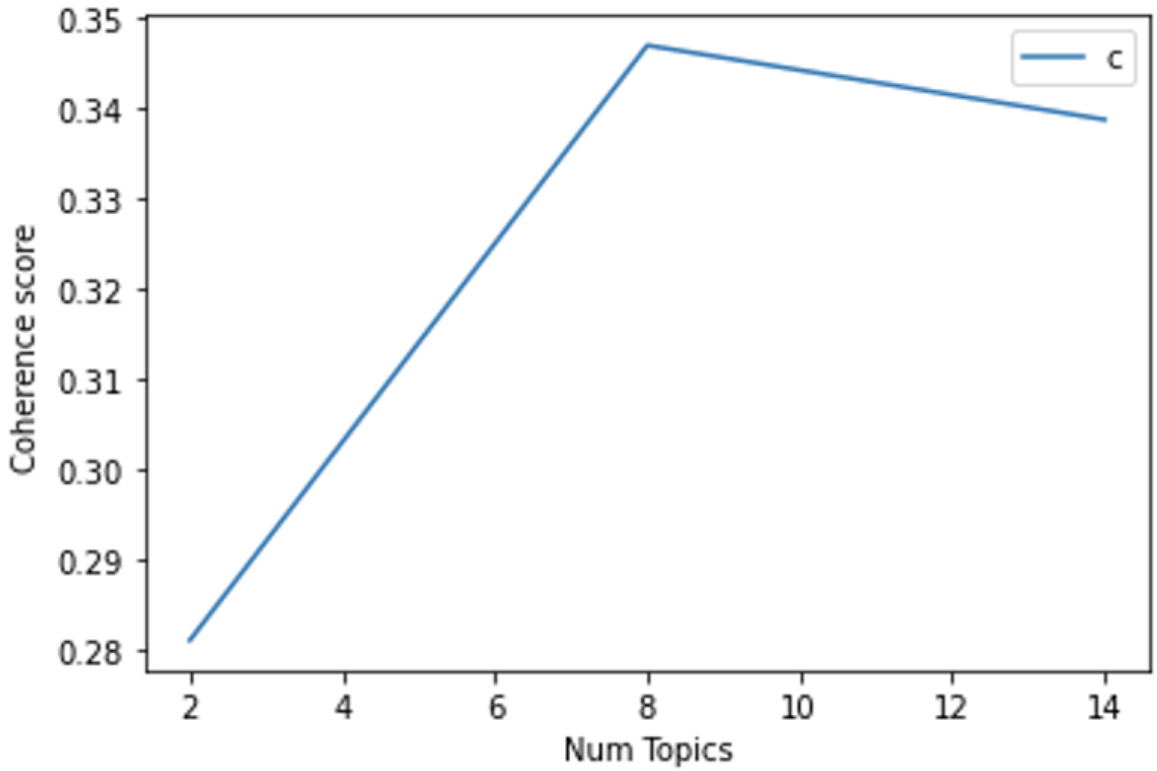
Saini, J. R. & Rakholia, R. M. (2016). On Continent and Script-Wise Divisions-Based Statistical Measures for Stop-words Lists of International Languages, Procedia Computer Science, Volume 89, pp. 313-319, ISSN 1877-0509, Available online: https://doi.org/10.1016/j.procs.2016.06.076 [Accessed 25th May 2022]

Sit, M. A., Koylu, C., & Demir, I. (2019). Identifying disaster-related tweets and their semantic, spatial and temporal context using deep learning, natural language processing and spatial analysis: a case study of Hurricane Irma, International Journal of Digital Earth, 12:11, 1205-1229, DOI: 10.1080/17538947.2018.1563219 [Accessed 28 May 2022]

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1 (January 2014), 1929–1958.

Vikramkumar, Vijaykumar, B., & Trilochan (2014). Bayes and Naive Bayes Classifier. ArXiv, abs/1404.0933. DOI: 10.1080/17538947.2018.1563219 [Accessed 28 May 2022]

Appendix A. Coherence score



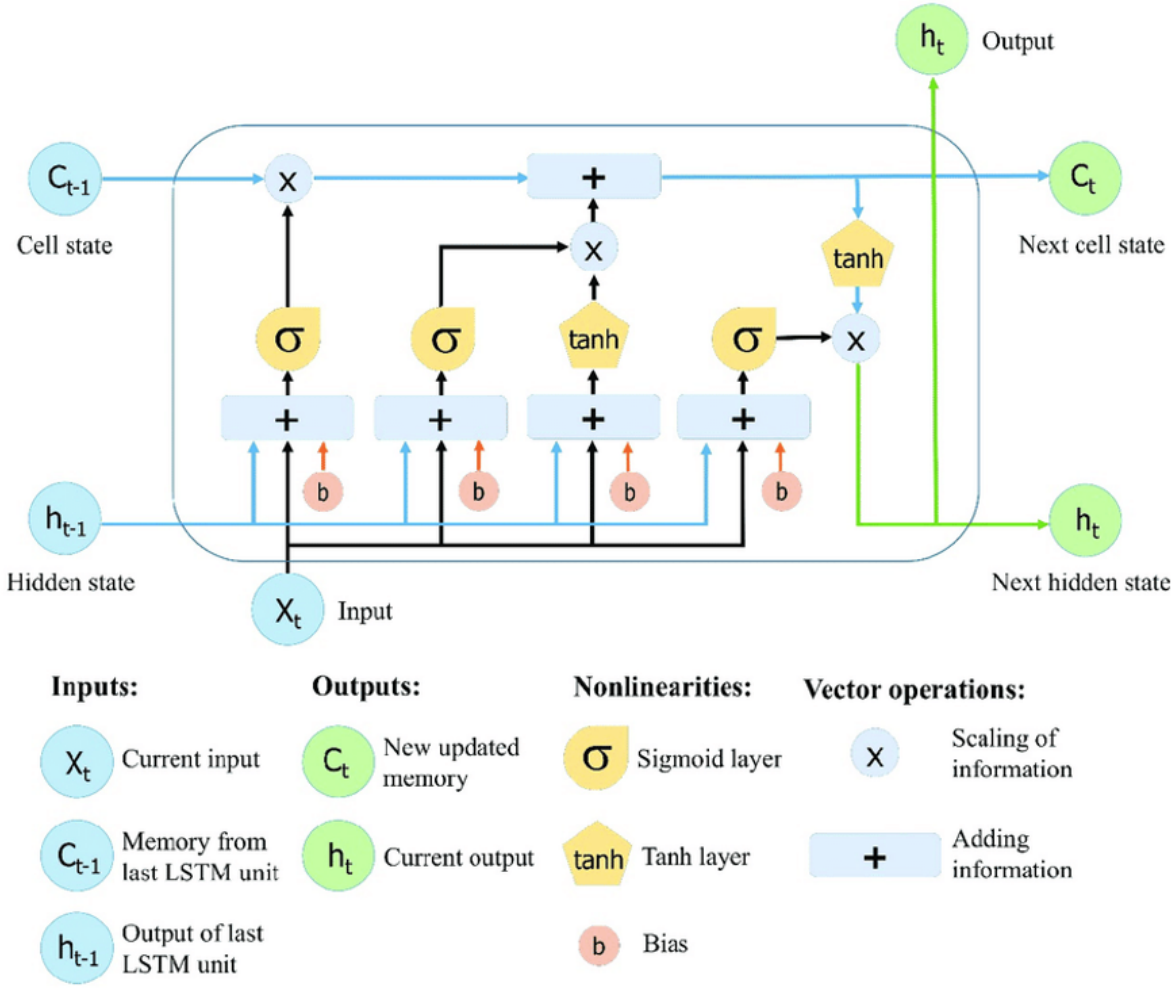
Appendix B. The results of the LDA model

```
[(0, '0.010*"get" + 0.009*"latest" + 0.009*"california" + 0.009*"wildfire" + 0.008*"amp" + 0.008*"home" + 0.008*"news" +
0.008*"watch" + 0.007*"nuclear" + 0.007*"weather"'),

(1, '0.020*"hiroshima" + 0.011*"fire" + 0.009*"bombing" + 0.008*"mass" + 0.007*"atomic" + 0.007*"japan" + 0.006*"murder" +
0.006*"school" + 0.005*"year" + 0.005*"wa"'),

(2, '0.013*"fire" + 0.013*"crash" + 0.012*"disaster" + 0.009*"obama" + 0.009*"wa" + 0.007*"via" + 0.006*"plan" + 0.006*"ha" +
0.006*"emergency" + 0.005*"say"'),

(3, '0.026*"fire" + 0.012*"building" + 0.009*"burning" + 0.008*"california" + 0.008*"migrant" + 0.008*"attack" + 0.008*"news" +
0.008*"video" + 0.007*"rescuer" + 0.007*"police"'),

(4, '0.012*"train" + 0.006*"û" + 0.005*"people" + 0.005*"flood" + 0.005*"home" + 0.005*"two" + 0.005*"fedex" + 0.005*"refugee" +
0.005*"life" + 0.005*"like"'), (

5, '0.009*"oil" + 0.008*"spill" + 0.007*"û" + 0.006*"possible" + 0.006*"may" + 0.006*"wa" + 0.005*"county" + 0.005*"projected" +
0.005*"bigger" + 0.005*"severe"'),

(6, '0.016*"family" + 0.014*"legionnaire" + 0.011*"fatal" + 0.010*"wa" + 0.009*"outbreak" + 0.008*"charged" + 0.008*"affected" +
0.007*"boy" + 0.007*"accident" + 0.006*"airplane"'),

(7, '0.018*"suicide" + 0.012*"bomber" + 0.011*"bomb" + 0.007*"saudi" + 0.007*"kill" + 0.007*"killed" + 0.007*"people" +
0.007*"dead" + 0.007*"mosque" + 0.006*"old"')]
```

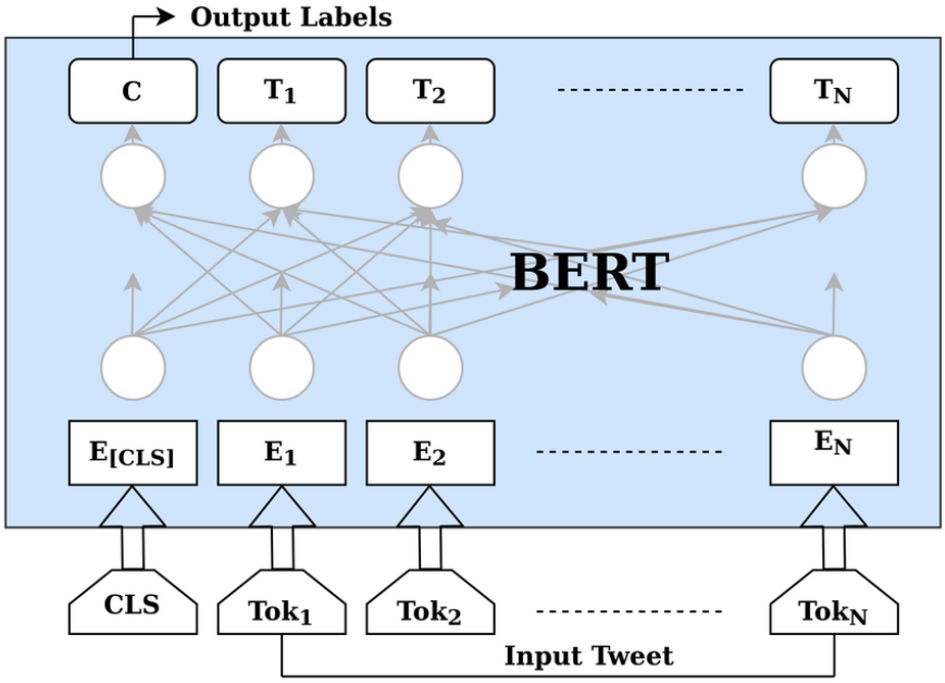
Appendix C. The results of the 8 largest topics of the Top2Vec model

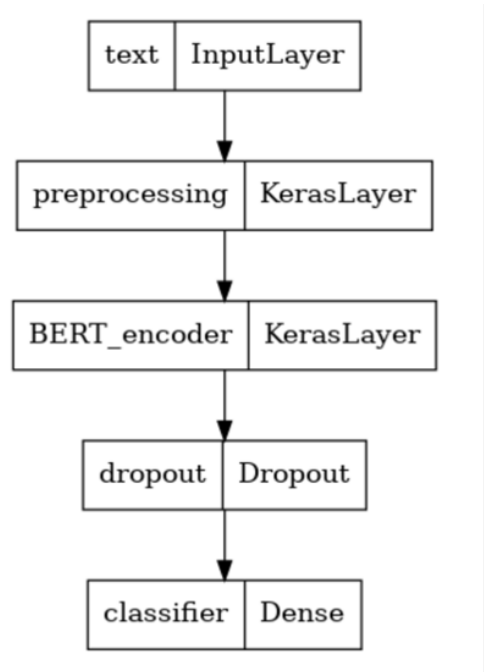
```
Top words for topic 35: [('death', 0.041143505586888465), ('collapse', 0.04073365625730158), ('like', 0.03884105333906134), ('wa',
0.034363440457253855), ('closed', 0.02788319109194826), ('ha', 0.0232689018404457), ('bridge', 0.021570748885873032),
('palestinian', 0.021470811120327636), ('fatality', 0.021237379202028336), ('dead', 0.02097117536492618)]

Top words for topic 8: [('hiroshima', 0.20484760560477694), ('nuclear', 0.16340398674981205), ('atomic', 0.11691837961353291),
('japan', 0.10991282667864698), ('bombing', 0.0955196530723299), ('iran', 0.07343463761651077), ('anniversary',
0.07007154355124022), ('mark', 0.06424790322454269), ('bomb', 0.057704777705110406), ('nagasaki', 0.05546313647843926)]

Top words for topic 39: [('mass', 0.10807277102609378), ('attack', 0.0891833669132357), ('police', 0.07534105244084262),
('murder', 0.06897097845693109), ('hostage', 0.06484366261565627), ('militant', 0.06053713211878946), ('murderer',
0.059423016234988274), ('terrorist', 0.059422227187621926), ('udhampur', 0.05339166807048927), ('massacre', 0.049687878151327906)]

Top words for topic 33: [('forest', 0.20170099467380231), ('california', 0.18303615613510676), ('wildfire', 0.1646254304422506),
('wild', 0.09958810128168764), ('bush', 0.06068619087814901), ('service', 0.05565650164468599), ('firefighter',
0.050571825731790844), ('rocky', 0.0454283718507652), ('burning', 0.04521053490103729), ('contained', 0.04309224048507481)]

Top words for topic 32: [('building', 0.24441110479022782), ('burning', 0.13477552203564994), ('flame', 0.08325960665251944),
('truck', 0.07751604271543323), ('manchester', 0.057172381620343965), ('street', 0.052514786980937235), ('dutch',
0.04568955641469667), ('wa', 0.04296110832417239), ('damaged', 0.042879286215257974), ('ablaze', 0.040950861790570885)]

Top words for topic 36: [('youtube', 0.12166223070035691), ('video', 0.058523912520392844), ('liked', 0.05632610498742969),
('hijack', 0.05357529083672099), ('patience', 0.04153928671784459), ('jonathan', 0.04153928671784459), ('bayelsa',
0.04153928671784459), ('amp', 0.04045312882701951), ('love', 0.03953050646824812), ('reddit', 0.03699327361247786)]

Top words for topic 34: [('suicide', 0.43454106258057923), ('mosque', 0.3755692236569571), ('saudi', 0.37190073772374793),
('bomber', 0.28454729880604795), ('bombing', 0.1881161041542056), ('kill', 0.1801295187744092), ('security', 0.1496890575036326),
('site', 0.13193024699455053), ('claim', 0.11798504961347148), ('reuters', 0.08839775143503754)]

Top words for topic 29: [('train', 0.4650495689428277), ('derailed', 0.17558812915552166), ('derail', 0.14531821616150034),
('derailment', 0.132990031452278), ('metro', 0.1071856826317835), ('rail', 0.10097245289830148), ('india', 0.0993861372654637),
('passenger', 0.08729296186789395), ('freakiest', 0.07531306822383504), ('line', 0.07489101469688614)]
```

Appendix D. LSTM node architecture (Le, Ho, Lee & Jung, 2019)



Appendix E. BERT Architecture (Gundapu & Mamidi, 2021)





Appendix F. Confusion Matrix