REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

IDENTIFYING AND CLASSIFYING DISASTER-RELATED TWEETS

18011094 — Toygar Tanyel

COMPUTER PROJECT

Advisor
Prof. Dr. Mine Elif KARSLIGİL

December 2021, January 2022

ACKNOWLEDGEMENTS

I would like to acknowledge and give my warmest thanks to my advisor Prof. Dr. Mine Elif KARSLIGİL who made this work possible. Her guidance and advice carried me through all the stages of writing my project.

Toygar Tanyel

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

BERT	Bidirectional Encoder Representations from Transformers
GloVe	Global Vectors for Word Representation
CNN	Convolutional Neural Networks
BiLSTM	Bidirectional Long Short-term Memory
GRU	Gated Recurrent Units
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
AP	After Pre-process
USGS	United States Geological Survey

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Identifying and Classifying Disaster-Related Tweets

Toygar Tanyel

Department of Computer Engineering
Computer Project

Advisor: Prof. Dr. Mine Elif KARSLIGİL

In this study we propose a competitive deep learning model pipeline for disaster analysis using tweets. Our context dependent and independent pipelines are developed to decrease problems in training phase, and to solve these problems, our model pipeline constructed as the combination of convolutional and sequence processing layer will be evaluated with different type of word embeddings. To feed our model we utilized the social media data. Social media has an important role in today's world especially at the moments of disaster related crisis. Social media generates an enormous amount of data and provides a special opportunity to classify disaster Tweets from regular ones. In particular microblogging platforms such as Twitter, are becoming approved among many researchers as a real-time scraping platform for getting actionable and tactical information during disasters. Due to its instantaneous speciality, more and more agencies are using Twitter for monitoring disaster events to plan a quick rescue. However, building an accurate predictive model to identify disaster Tweets quite challenging. The lack of sufficient context due to the length limit reveals many other problems. Also, it can be difficult to distinguish between disaster-related Tweets and regular Tweets due to word ambiguity. For disaster analysis task, we acquired promising results with proposed model pipeline.

Keywords: Social media, natural language processing, deep learning, text classification, text pre-processing.

ÖZET

Afetle İlgili Tweetleri Belirleme ve Sınıflandırma

Toygar Tanyel

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Prof. Dr. Mine Elif KARSLIGİL

Bu çalışmada, tweetleri kullanarak afet analizi yapmak için rekabetçi bir ardışık derin öğrenme modeli öneriyoruz. Eğitim aşamasındaki sorunları azaltmak için bağlam bağımlı ve bağımsız iki ardışık modelimiz incelenmiş ve gerçeklenmiştir. Bu model, evrişimsel sinir ağları (CNN) ve iki yönlü uzun-kısa vadeli bellek(BiLSTM) ardışık olarak farklı tipteki (GloVe & BERT) kelime vektörleri ile etkili sonuç verebilecek şekilde tasarlanmıştır. Modelimizi beslemek için sosyal medya verilerini kullandık. Günümüzde, özellikle afet kaynaklı kriz anlarında sosyal medyanın önemli bir yeri vardır. Sosyal medyanın ürettiği yüksek miktardaki veri, afet analizi için eşsiz bir fırsat sunmaktadır. Özellikle Twitter gibi sınırlı sayıdaki karakterle bilgi veren platformlar, afet sırasında eyleme geçirilebilir ve taktiksel bilgi edinimi için gerçek zamanlı bir bilgi çıkarım platformu olarak birçok araştırmacı tarafından aktif olarak kullanılıyor. Anlık bilgi alınabilme özelliği nedeniyle, daha fazla kurum, hızlı bir kurtarma planı yapmak için afet olaylarını izlerken Twitter'ı kullanıyor. Ancak, uzunluk sınırı nedeniyle yeterli bağlamdan yoksun olabilecek felaket Tweetlerini belirlemek için doğru bir tahmine dayalı model oluşturmak zordur. Bununla birlikte bu modeli destekleyecek veri seti oluşturmak da aynı oranda zorlu bir görevdir. Afet analizi için önerdiğimiz model ile umut verici sonuçlar elde ettik.

Anahtar Kelimeler: Sosyal medya, keşifsel veri analizi, sınıflandırma, derin öğrenme, BERT, GloVe, CNN, BiLSTM.

1 Introduction

In this study we provide a model pipeline which includes combination of deep learning models. The model offers high-quality linguistic features extraction even in deficient datasets. Our deep learning model pipeline developed by 3 different modules, an embedding layer (BERT or GloVe), a 1D convolutional layer, and a bidirectional lstm layer.

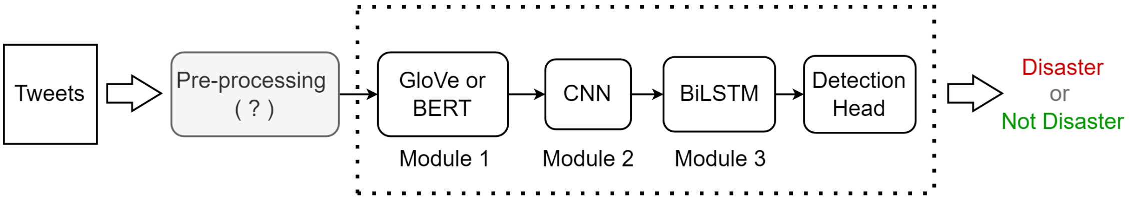


Figure 1.1 The proposed learning pipeline. (?) indicates whether the dataset has been preprocessed or not. Module 1 is embedding layer, Module 2 is a convolutional layer, and Module 3 is a bidirectional lstm layer. Detection head determines the result.

We utilized natural language processing methods and models to accelerate progress of improvements on the task. Pre-processing step is one of our study to show its effects with different model pipelines. Therefore, this step both performed and unperformed. The GloVe and BERT embeddings are chosen for extracting, respectively, statistical and sentimental features from texts to show difference between them. CNN is employed as feature extractor of the model pipeline while BiLSTM is keeping the chronological order between words in given texts. The last layer which named detection head shows up the result.

1.1 Motivation

We offer classifying disaster tweets from regular ones, and creating a system where provides locations or essential information to relevant institutions and organizations is significant necessity during disasters such as earthquakes, floods and forest fires.

In this case, social media platforms which can be considered as instantaneous information sources are remarkable to utilize for good purpose.

During a time of crisis, people tend to use social media platforms to post situational updates, look for useful information, and ask for help. Twitter has reached enormous popularity since its beginning. The latest usage statistics of Twitter data show that as of May 2020, on average, around 6,000 tweets are sent in every second. Society generates large volumes of social data that is used by many high-level analytics practices to create additional value. Moreover, many studies have utilized Twitter data to implement natural language processing practices such as sentiment analysis, named entity recognition, and topic modelling.

In addition to its social function, Twitter increase its popularity as being real time platform for tracking events, including disasters, accidents and emergencies, and collecting useful and legal analytical data for policy or marketing. Information becomes knowledge, especially among the new generations which its majority have a smartphone where allows everyone to share an emergency Tweet instantly to be seen. More and more agencies such as disaster relief organizations and news agencies are starting to realize that there are new ways to reach people to channel resources due to the convenience of social media interaction. Organizations constantly monitor Twitter. Thus, first responders can be deployed and rescue plans can be drawn up as soon as possible.

For instance, as given in [1] the first report of the Westgate Mall attack in Nairobi, Kenya in 2013 was published on Twitter, almost 33 minutes before a local TV channel reported the event. Likewise, the news about the Boston bombing incident appeared on Twitter before any other news channel reported the event. Similarly, in the case of the California earthquake it was observed that the first half dozen tweets were recorded by Twitter about a minute earlier than the recorded time of the event according to the USGS. These direct reports come from witnesses and spectators, those who are directly observing what is occurred.

However, the automation of the process requires an robust and quite accurate classifier to distinguish disaster-related tweets from regular ones. The disaster prediction based on tweets is difficult due to people who can use metaphorical words to describe other things.

2
Related Work

Twitter is becoming popular platform which allows researchers to utilize the social data for emergency and disaster analysis in last few years [2–4]. Various methods are studied and implemented for extraction of key phrases for general purpose [5], disaster-relevant [6], and knowledge based events [7].

The latest works show that in disaster classification using fine-tuned BERT embeddings [8] is outperforming the old methods such as GloVe, Word2Vec and FastText [9],[10]. Moreover, intuitive reason of outperforming is the GloVe like models generate embeddings that are context-independent, whereas BERT embeddings are context-dependent [11], however, it may depend on the dataset [12]. Meanwhile, Bi-LSTM, CNN and GRU can be considered as some of the most common machine learning models in classification tasks. Recently, some studies focuses on creating a hybrid models that include CNN-BiLSTM pipeline with using different embeddings [8], [13] to propose state-of-art results. Besides of these, [10] propose a valuable priority scheduling algorithms to plan rescues with remarkable results.

3
Feasibility Report

In this study we offer a deep learning model pipeline which requires high hardware and software resources to classify disaster-related tweets. GloVe and BERT are used to obtain dependent and independent word embeddings for training phase. These embeddings utilize CNN to extract features and BiLSTM to keep order of sequential text data. The pipeline needs to be fed by more than one hundred thousand sentences to reach competitive results for real world applications. Moreover, the amount of data expects large enough disk space from user. Meanwhile, various kind of deep learning libraries will be used in the project to achieve project goals. Creating a competitive deep learning model for disaster prediction comes with requirement of high level feasibility report to understand the system is necessary or not.

3.1 Technical Feasibility

This study focuses on the technical resources available to the organization. It helps organizations determine whether their technical resources meet the capacity and whether the technical team has the ability to implement ideas into working systems. Technical feasibility also includes evaluation of hardware, software and other technical requirements of the proposed system.

3.1.1 Software Feasibility

Related works show that Python is the dominant language for artificial intelligence tasks. Therefore, project will be constructing through Python and its libraries which provide users easier implementation due to enormous open-source contribution by Python community.

The basic libraries used in the project are as follows:

- **NumPy:** NumPy is the essential package for scientific computing with Python.

Define N-dimensional arrays and matrices, and work on these arrays and matrices. It is a Python library where mathematical functions can be applied.

- **Pandas:** Pandas is a Python package that provides fast and flexible data structures designed to make it easy to work with "relational" or "labeled" data. It is also a library with very powerful filtering functions to clean data.
- **scikit-learn:** It is a Python library that contains many algorithms for Machine Learning and allows us to easily manipulate the results of algorithms.
- **Tensorflow:** Tensorflow is an open source deep learning library.
- **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Operating system considerations start with Linux since it is open-source and free. However, it is not favourable for some users, the process of implementation include more complexity and user action than others. MacOs, on the other hand, is not strategic due to project owner do not have an Apple computer and it would be costly to purchase. Windows operating system will be preferred in the project due to its generalizability and easy to use.

PyCharm, Spyder, Jupyter Notebook, Google Colaboratory and Kaggle Notebooks are environments that available to implement project. These can be used as cross-platforms due to their own special features. IDE choice does not affect the results of the project. Besides these, CSV file format will be used to handle with dataset.

3.1.2 Hardware Feasibility

To develop the optimum and sustainable system, one computer with Windows operating system, minimum 16GB RAM memory, 50GB+ disk space, 2080Ti GPU, and Intel Core i7-10700K are required.

As a student, project developed by the system with Intel Coffee Lake Core i7-8750H CPU, 4GB GDDR5 Nvidia GTX1050Ti 128 Bit GPU, and 8GB (1x8GB) DDR4 2666MHz RAM.

To store information, cloud services are convenient to choose. However, for massive companies and organizations which have private information on their clouds might not tend to use cloud services on this purpose. Thus, the main disks with 50GB+ place are appropriate to avoid problems with companies. In both case, system is available to maintain.

As a student, project will be implemented through Google Drive due to its smooth data connection with Google Colaboratory. Google Drive is convenient to store dataset, generated word embeddings and results. There is no cost until reaching to 15GB Cloud limit. However, there is usage time limit for GPU and TPU of Google Colaboratory. As words of Google, *Notebooks run by connecting to virtual machines that have maximum lifetimes that can be as much as 12 hours. Notebooks will also disconnect from VMs when left idle for too long. Maximum VM lifetime and idle timeout behavior may vary over time, or based on your usage. This is necessary for Colab to be able to offer computational resources for free.*

To generate word embeddings and training, Google Colaboratory will be used. However, as backup plan if we are faced problem with Google Colaboratory, Kaggle will be used due to their free cloud GPU and TPU offers. Meanwhile, depending on the local features, 1080TI GPU or 2080TI GPU is convenient to train in local computer.

3.2 Social Feasibility

The users of the system will be employees/volunteers of organizations such as civil protection organizations and news agencies. The system offers an easy-to-use code interface.

As another way of utilization, researchers and curious people have a chance to improve the system by their unique ideas.

3.3 Management Feasibility

The system provides the information where disaster relief organizations and news agencies can use in their rescue plans. Users from the agencies will have authorization to change their local system due to their requirements.

3.4 Legal Feasibility

License fees to be paid for basic system requirements, server rental/purchase costs and the expenses of the staff who will work in this system will be covered through an official channel and legal obligations will be fulfilled.

The data which is used in the system is public on Kaggle Competition and includes no private information.

The system has been established in a unique and purpose-oriented structure where it can perform its own basic functions. If the system conflicts with a business agreement and requires special permits, licenses, or structural changes, these can be edited and corrected in real time if these cannot be provided before the end of the project.

3.5 Economic Feasibility

The project does not constitute a legal obligation since the Python software and packages used in project are free. At the same time, open-source data from Kaggle, again, free of charge, and Python & libraries can provide licenses that allow visualizations without any financial obligation for everyone.

Possible scenarios, economic calculation:

$X \longrightarrow$ *Amount of Employee/Volunteer who will use the software*

Table 3.1 Expenses

Windows license cost	$X * 1.300$TL
Equipment cost	$X * 6.000$TL
Colab Pro cost	120\$ $*$ 12months
Salary of project owner	750TL
Total = $(7.300 * X)$TL + 19.650TL	

$Xf \longrightarrow$ *Amount of Employee/Volunteer who will be in the field*

$Y \longrightarrow$ *Amount of Events*

Table 3.2 Revenues

Preventing from misdirection	$Y * 5.000$TL
Workforce decrease cost	$Xf * 3.500$TL
Saved resources	$Y * Xf * 15.000$TL
Total = $((5.000 * Y) + (3.500 * Xf) + (15.000 * Xf * Y))$TL	

Net Return Ratio:

$$[(5.000 * Y) + (3.500 * Xf) + (15.000 * Xf * Y) - ((7.300 * X) + 19.650)] \text{ TL}$$

3.6 Time Feasibility

An undergraduate student worked on the project and it was completed within one academic term has been planned. The project consists of three main processes. The first of these, organize the project pipeline and start literature review, secondarily, determine and collect proper data for the project. In conclusion, getting results through different machine learning algorithms.

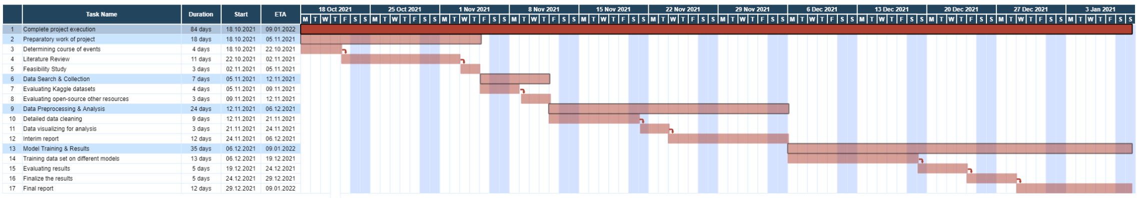
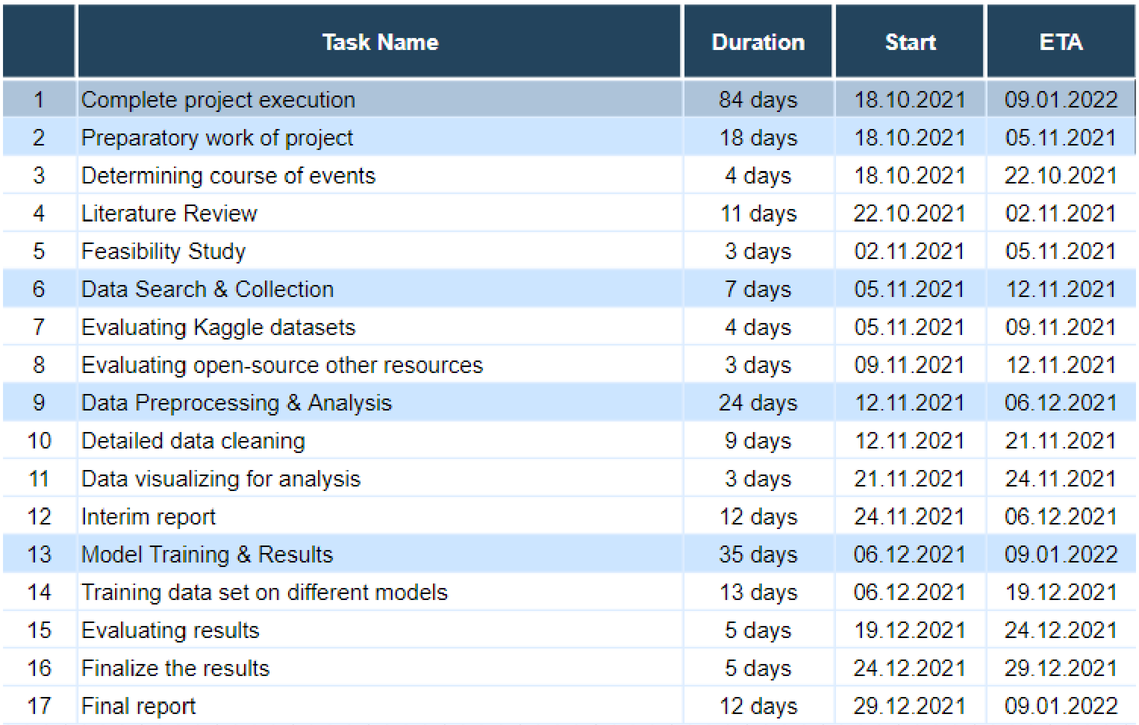


Figure 3.1 Gantt Diagram



	Task Name	Duration	Start	ETA
1	Complete project execution	84 days	18.10.2021	09.01.2022
2	Preparatory work of project	18 days	18.10.2021	05.11.2021
3	Determining course of events	4 days	18.10.2021	22.10.2021
4	Literature Review	11 days	22.10.2021	02.11.2021
5	Feasibility Study	3 days	02.11.2021	05.11.2021
6	Data Search & Collection	7 days	05.11.2021	12.11.2021
7	Evaluating Kaggle datasets	4 days	05.11.2021	09.11.2021
8	Evaluating open-source other resources	3 days	09.11.2021	12.11.2021
9	Data Preprocessing & Analysis	24 days	12.11.2021	06.12.2021
10	Detailed data cleaning	9 days	12.11.2021	21.11.2021
11	Data visualizing for analysis	3 days	21.11.2021	24.11.2021
12	Interim report	12 days	24.11.2021	06.12.2021
13	Model Training & Results	35 days	06.12.2021	09.01.2022
14	Training data set on different models	13 days	06.12.2021	19.12.2021
15	Evaluating results	5 days	19.12.2021	24.12.2021
16	Finalize the results	5 days	24.12.2021	29.12.2021
17	Final report	12 days	29.12.2021	09.01.2022

Figure 3.2 Gantt Diagram - Processes

4
System Analysis

The system offers a rapid prediction using deep learning model pipeline. The promptness and quick dispatch is crucial in the time of disaster. Therefore, we created a system where provides reasonable accuracy for classifying disaster tweets using GloVe-CNN-BiLSTM and BERT-CNN-BiLSTM pipelines.

Text classification is the most fundamental and essential task in many natural language processing applications, and has extensive history [14]. The system of classifying disaster-related tweets requires special attention due to its critical role. Results should be as accurate as possible to reduce false resource allocation.

The classification system based on sustainable training process with extendable data. In this case the system offers two ways of usage.

First action is that the user should initially determine the text of disaster tweets. After this selection, the text data information of the tweet is given directly to ready model and display the results with its confidence. Second action is that the user which is technical person should gather tweets without any pre-processing except labeling. The system will handle cleaning of the new data after combination with the old one.

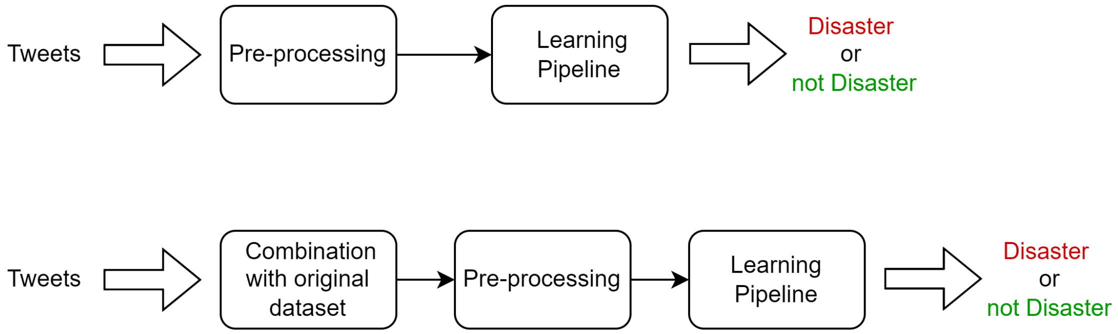


Figure 4.1 The proposed system overview for disaster prediction using tweets. First and second action sequentially.

The disaster classification system has features which are called hyperparameters. These features shape the learning process in many important ways. The period of training, accuracy, and convergence depends on the hyperparameters. The parameters were kept secret to prevent the user from divergence during the backprop stage. Convenient parameters have already choosen for original data. If the technical user has high intuition that hyperparameter changes should be made, she/he will have to reach the project owner.

The analysis of the selected tweets will begin with the new model that has been prepared. The result will be shown in seconds due to importance of fast assistance dispatch to needy people. The results will be summarized in terms of disaster type, location and what is needed. User will check the results and choose the type of aid.

4.1 Use Case Diagram

Use case diagram is given as Figure 4.2 below.

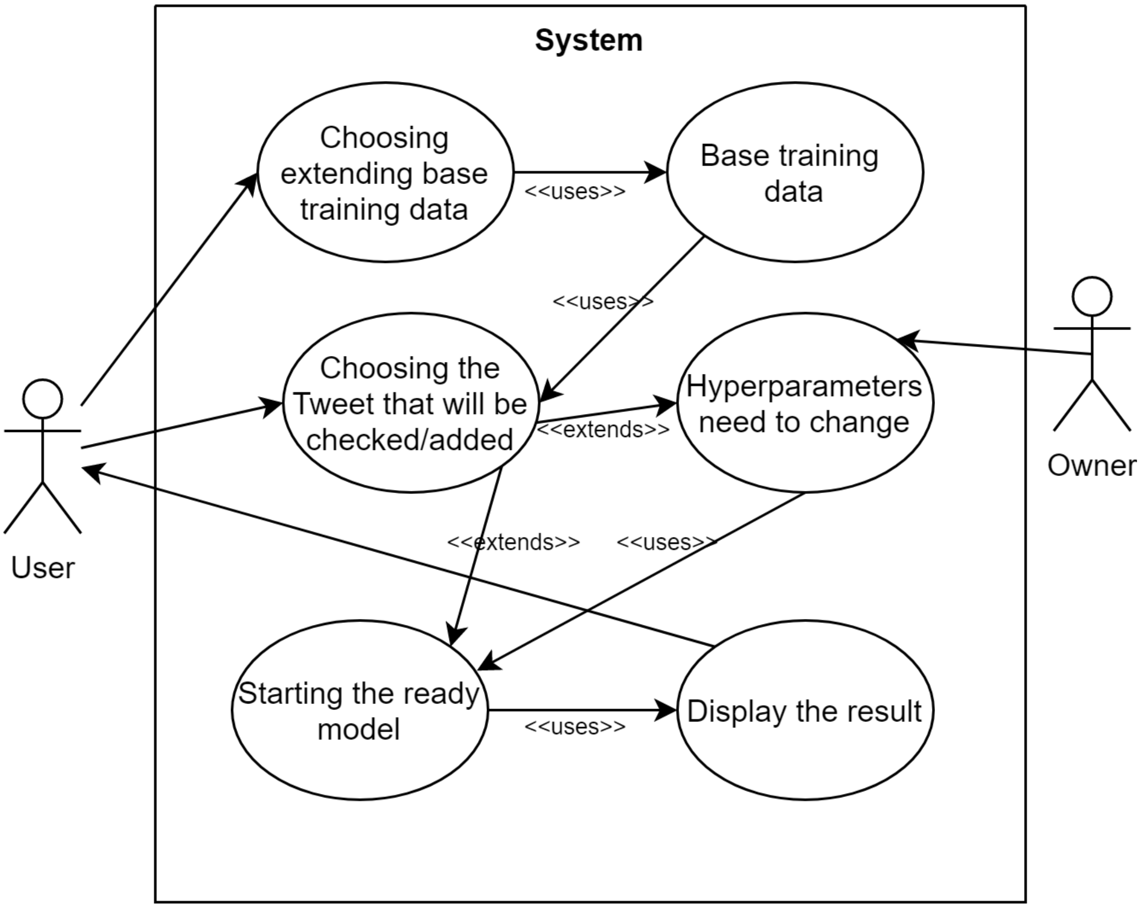


Figure 4.2 Use Case Diagram

5
System Design

The recent works demonstrate that in disaster classification using fine-tuned BERT embeddings is outperforming the old methods such as GloVe, and Word2Vec. However, it may depend on the dataset and combinations. Due to flexibility of model accuracy, both GloVe and BERT embeddings will be considered in separate designs. In machine learning, BiLSTM and GRU find extensive place in NLP classification tasks. The system runs through several instructions. **(1)** Words transform to embedding vectors. **(2)** Embeddings move into deep learning model. **(3)** System trains the model. **(4)** Display of the output.

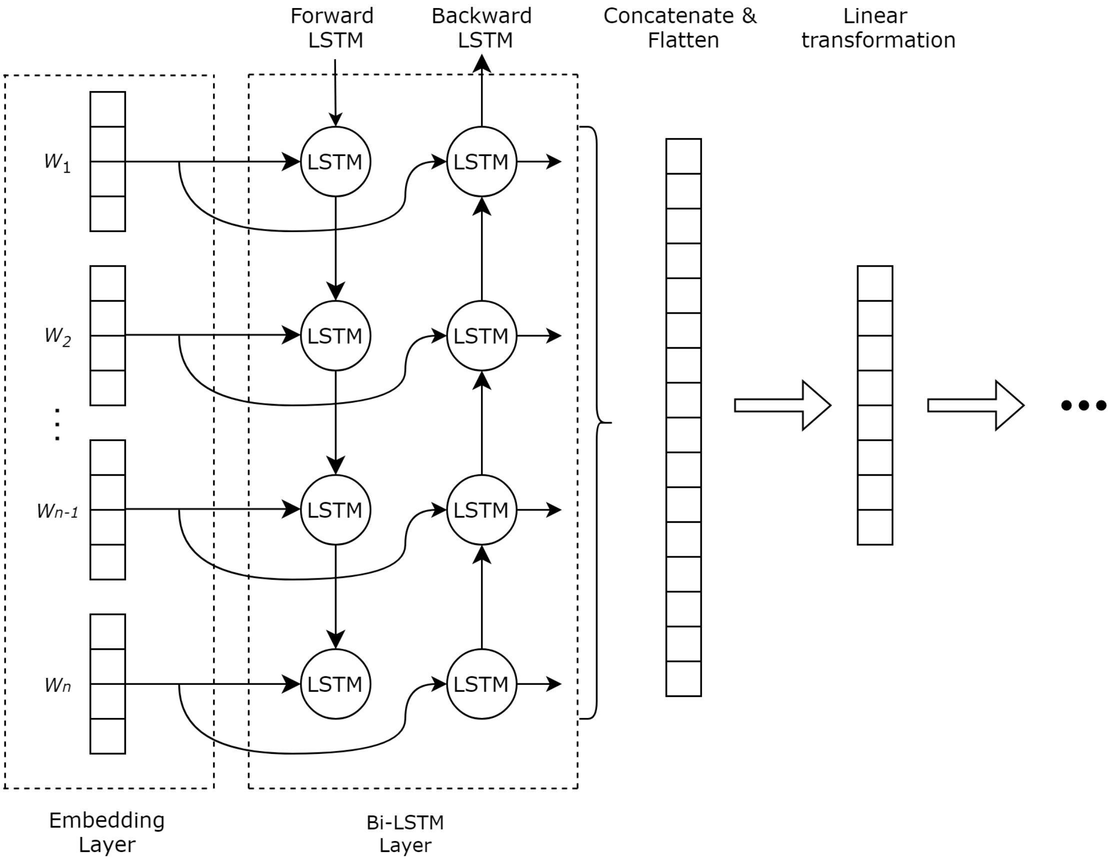


Figure 5.1 The embeddings and BiLSTM model architecture.

5.1 Word Embeddings

Word embedding is a learning expression of text in which words with the same meaning have similar expressions. In other words, word embedding is a term used to describe a word for text analysis and is usually in the form of a real-valued vector that encodes the meaning of the word, so the meaning of a word close to vector space is expected to be similar. In NLP tasks, any machine learning or deep learning models cannot use words or characters directly. Thus, word embedding approaches might be considered as breakthrough in deep learning on challenging NLP tasks.

5.1.1 GloVe

The unsupervised algorithm is based on data statistics. Models such as Skipgram and CBOW collect semantic information, but do not use co-occurrence statistics. Although matrix decomposition methods use these statistics, they cannot capture semantic relationships. There is no semantics in such models. The "GloVe" model proposed by Pennington et al. [15] aims to solve this problem by constructing a new objective function using probability statistics.

Mathematical perspective of GloVe:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})(w_i^T w_j + b_i + b_j - logX_{i,j})^2$$

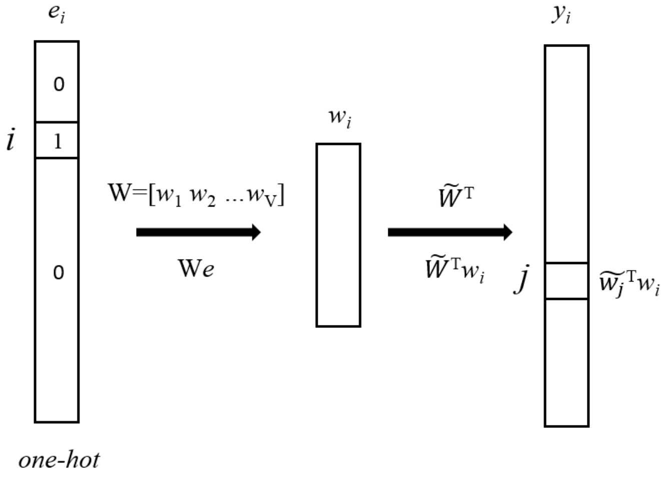


Figure 5.2 The model architecture of GloVe. The input is a one-hot representation of a word. The word embedding matrices serve as weight matrices in the model and thus the output of the model is a vector of inner products of word vectors. From [16].

5.1.2 BERT

In the *Attention is all you need* [17], Google introduced a new neural network architecture called transformer, which has many advantages over traditional sequential models (*RNN, GRU, LSTM etc.*). Advantages included, but were not limited to, more efficient modeling of long-term dependencies between tokens in a temporal order, and more efficient training of the model overall by eliminating sequential dependency on previous tokens.

BERT [18] is a new paper published by researchers in the Google AI language. Presenting cutting-edge results with various NLP assignments has had a significant impact on the deep learning community. Also, BERT uses masked-language modeling method and next sentence prediction task in training process to capture contextual information at the word and sentence level. To make things clearer, it's important at this point to understand the special tokens that the BERT writers used for fine-tuning and specific task training.

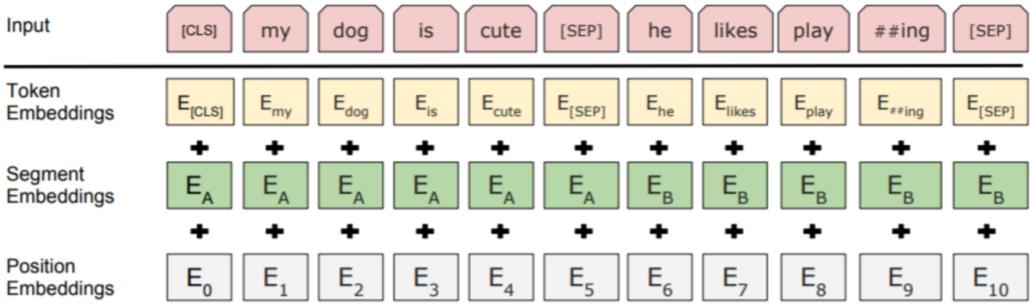


Figure 5.3 Input representation of BERT. The input is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

These tokens are the following: **[CLS]**, **[SEP]**, and **[MASK]**. In short, **[CLS]** is the first token of every sequence. **[SEP]** is the sequence delimiter token used in the pre-training of the sequence pair task. **[MASK]** is token used for masked words. It is used for pre-training only. The **input** layer is the vector of the sequence tokens along with the special tokens. The **Token Embeddings** are the lexical IDs for each of the tokens. The **Segment Embeddings** is a numeric class to distinguish between sentence A and B. Finally, the **Position Embeddings** refer the position of each word in the sequence.

5.2 Models

A classification model tries to draw some conclusion from the input values given for training. Given one or more inputs a classification model will try to predict the value of one or more outcomes. Outcomes are labels that can be applied to a dataset. For instance, *disaster* or *not disaster* which can be labelled as 1 and 0. Machine learning literature has lots of traditional ways to realize classification task such as random forest, support vector machine (SVM), naive bayes and logistic regression. However, to accomplish more complex tasks in deep learning, Bi-LSTM, CNN and GRU are some of the most common models that considerably convenient for classification tasks. In this paper we will analyse further only Bi-LSTM and CNN.

5.2.1 Bi-LSTM

Original LSTM [19] (Long Short-term Memory) previously published at 1997 to solve problem which called back-propagation through time. To be more clear, learning to store information over extended time intervals by recurrent back-propagation takes a very long time, mostly because of insufficient, decaying error back-flow.

A Bidirectional LSTM, is a sequence processing model that consists of two LSTM, one taking the input in a forward direction, and the other in a backwards direction. Bi-LSTM effectively increase the amount of information available to the network, improving the context available to the algorithm. For further information go through the original LSTM paper.

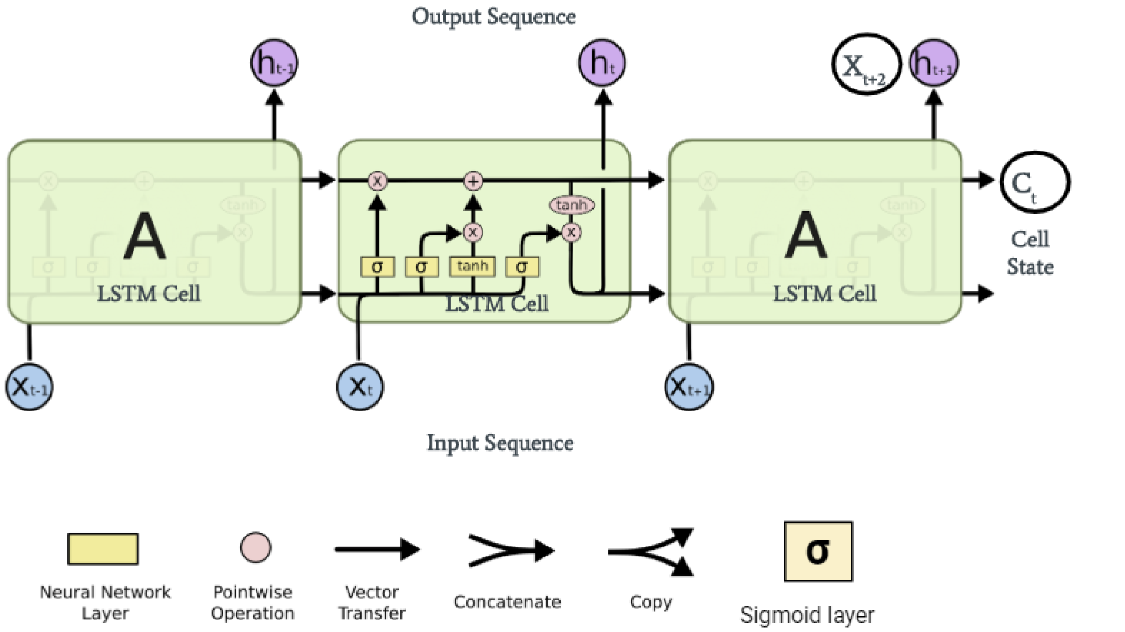


Figure 5.4 The LSTM cell architecture.

5.2.2 CNN

CNN is a class of deep, feed-forward artificial neural networks (where connections between nodes do not form a cycle) & use a variation of multilayer perceptrons designed to require minimal preprocessing. [20] shows that a simple CNN with little hyperparameter tuning and static vectors achieves remarkable results on multiple benchmarks. Learning task-specific vectors through fine-tuning offers further gains in performance.

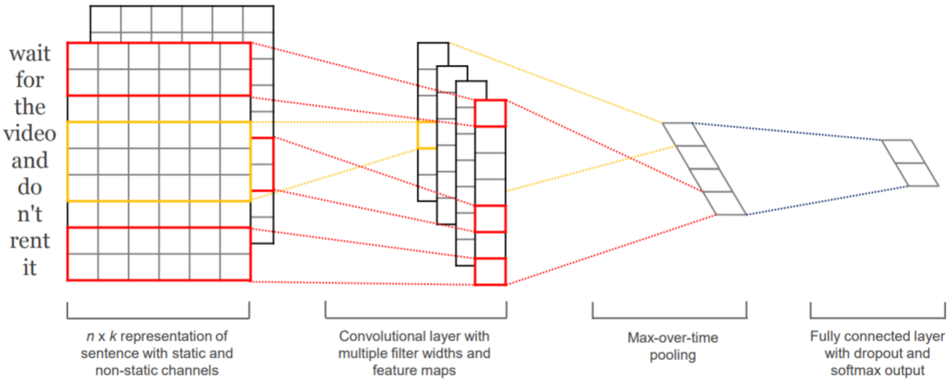

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 5.5 Model architecture with two channels for an example sentence.

5.3 Data Collection

In data collection process, labeling is quite time-consuming when the available time of project has considered. In other words, labeling requires large amount of time and workforce for one person. Therefore, the legal open-source datasets are searched and one of the dataset has been decided for the project in order to keep up with deadline. However, the optimum system requires more than one hundred thousand labeled data to achieve efficient accuracy.

5.4 Overview Of The Dataset

The chosen dataset contains Twitter data, and available as a Kaggle competition which is still in progress at here (accessed data on 28 Nov 2021). The dataset contain 10.876 samples. There are 7613 rows and 5 columns in train, and 3263 rows and 4 columns in test. The reason of difference with columns is that test set has no labels. The training set includes 3271 real and 4342 not real labels in terms of class distribution. Due to test set has no label, the train set is used as train-val-test set which divided to 6165 samples as train, 762 samples as dev, and 686 samples as test set to obtain results.

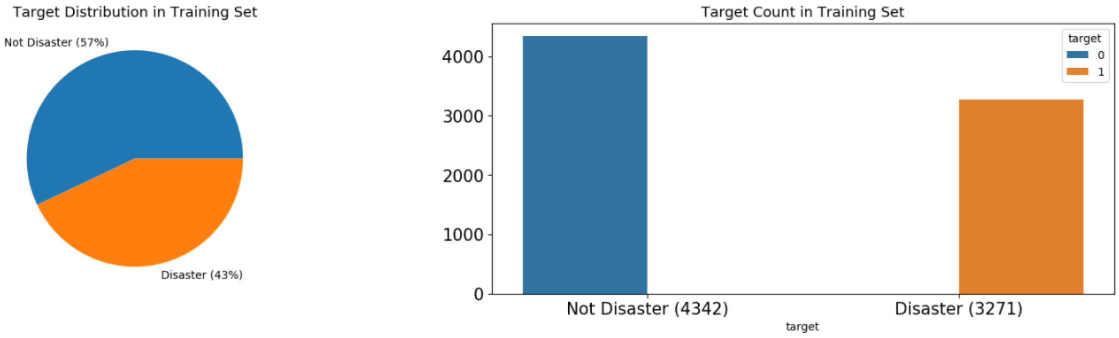
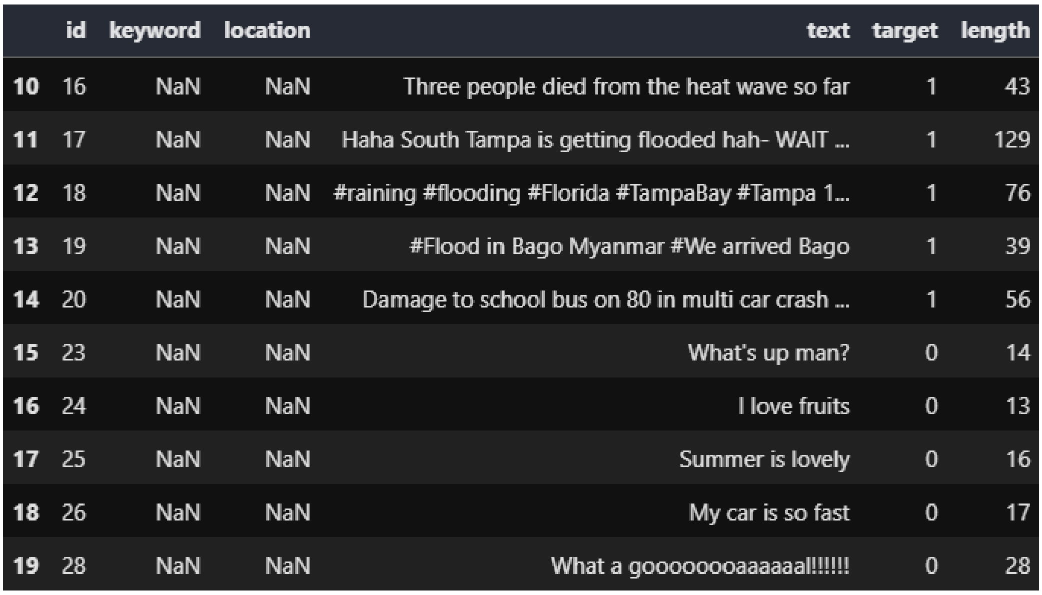


Figure 5.6 Training set distribution and target count.



	id	keyword	location	text	target	length
10	16	NaN	NaN	Three people died from the heat wave so far	1	43
11	17	NaN	NaN	Haha South Tampa is getting flooded hah- WAIT ...	1	129
12	18	NaN	NaN	#raining #flooding #Florida #TampaBay #Tampa 1...	1	76
13	19	NaN	NaN	#Flood in Bago Myanmar #We arrived Bago	1	39
14	20	NaN	NaN	Damage to school bus on 80 in multi car crash ...	1	56
15	23	NaN	NaN	What's up man?	0	14
16	24	NaN	NaN	I love fruits	0	13
17	25	NaN	NaN	Summer is lovely	0	16
18	26	NaN	NaN	My car is so fast	0	17
19	28	NaN	NaN	What a goooooooaaaaaal!!!!!!	0	28

Table 5.1 10 Samples from training set

At table 4.1, text length is added manually to understand difference between disaster and non-disaster tweets. The statistics show that non-disaster tweets are comparatively shorter than disaster ones. Moreover, the keyword and location are not available for most of the tweets which lead to ambiguity to determine priority.

5.4.1 Meta Features

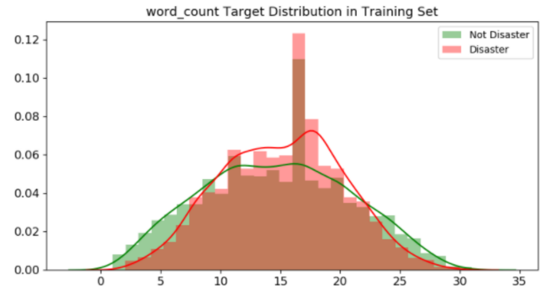
The distribution of class and record meta features helps identify disaster tweets. Most disaster tweets come from the news agencies. Therefore, disaster tweets seem to be more formalized in longer words than non-disaster tweets. Disaster tweets contain less typos than non-disastrous ones which comes from individuals.

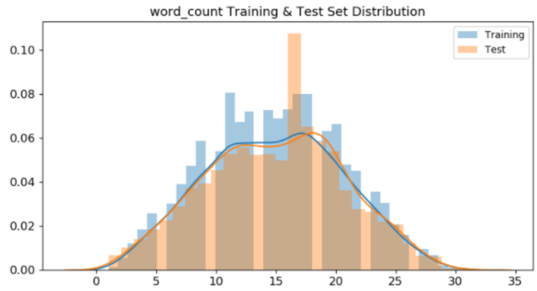
The following notations are used in the figures as meta features:

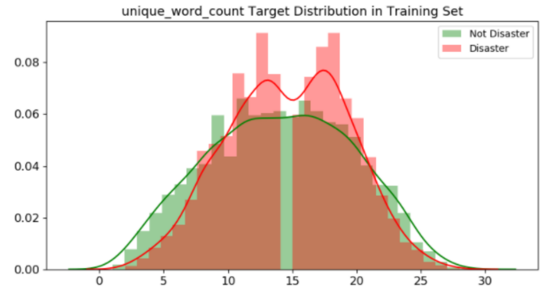
- ***word_count***: count of words in the text
- ***unique_word_count***: count of unique words in the text
- ***stop_word_count***: count of stop words in the text
- ***punctuation_count***: count of punctuations in the text
- ***hashtag_count***: count of hashtags (#) in the text
- ***mention_count***: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

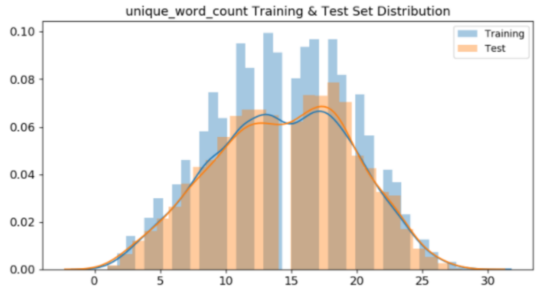
All the hyperfeatures have quite similar distributions in both training and test sets. It also indicates that the training set and the test set are taken from the same sample. In addition, meta features contain information about the target. However, some of them are not enough to give a hunch like url_count, hashtag_count and mention_count.

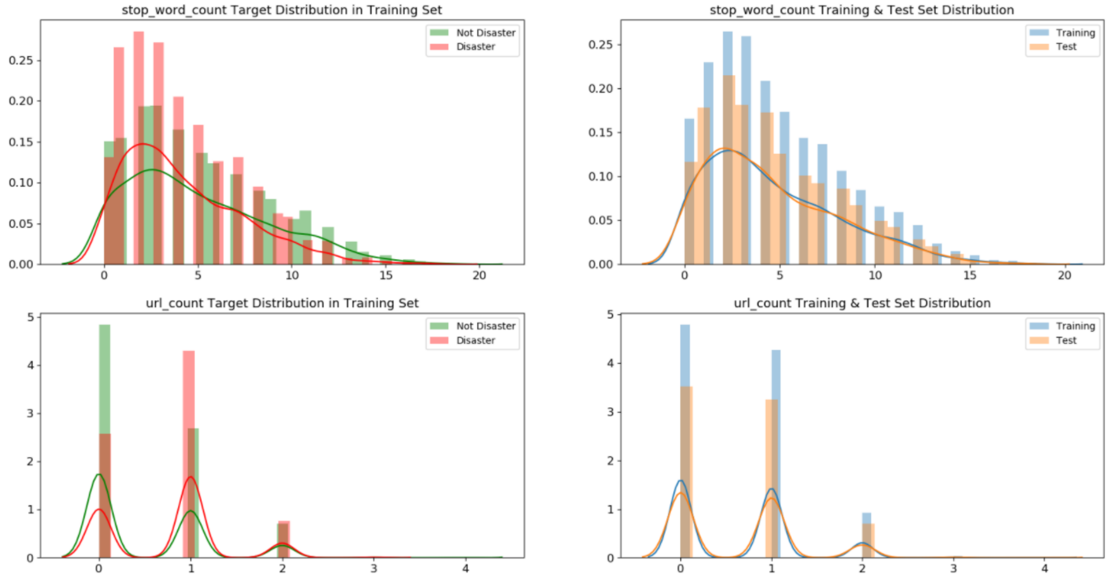
On the other side, word_count, unique_word_count, stop_word_count, mean_word_length, char_count, and punctuation_count have quite different distributions for disaster and non-disaster tweets. These features may be useful within models.

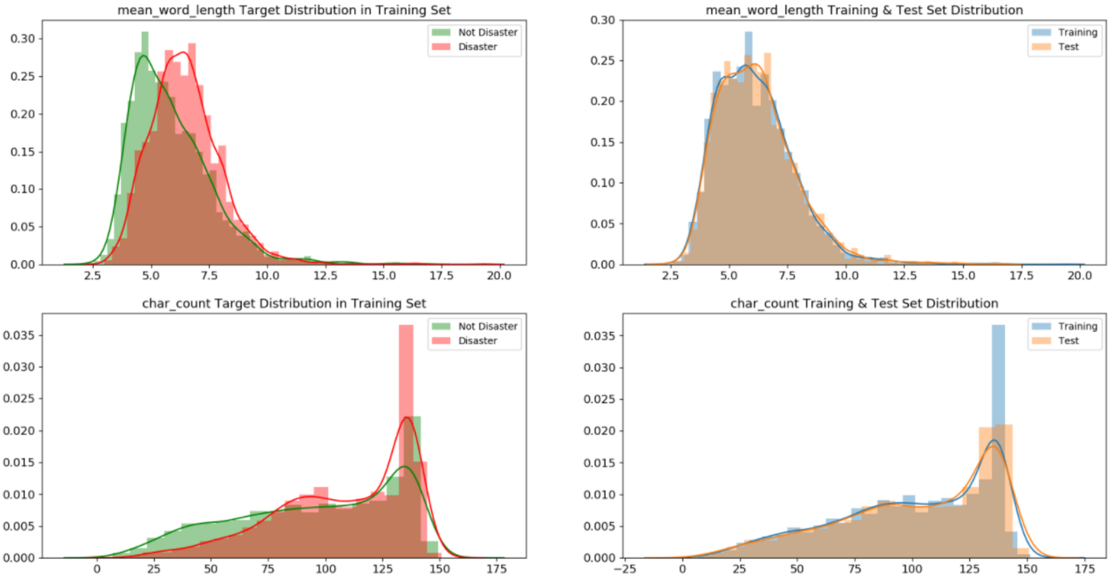


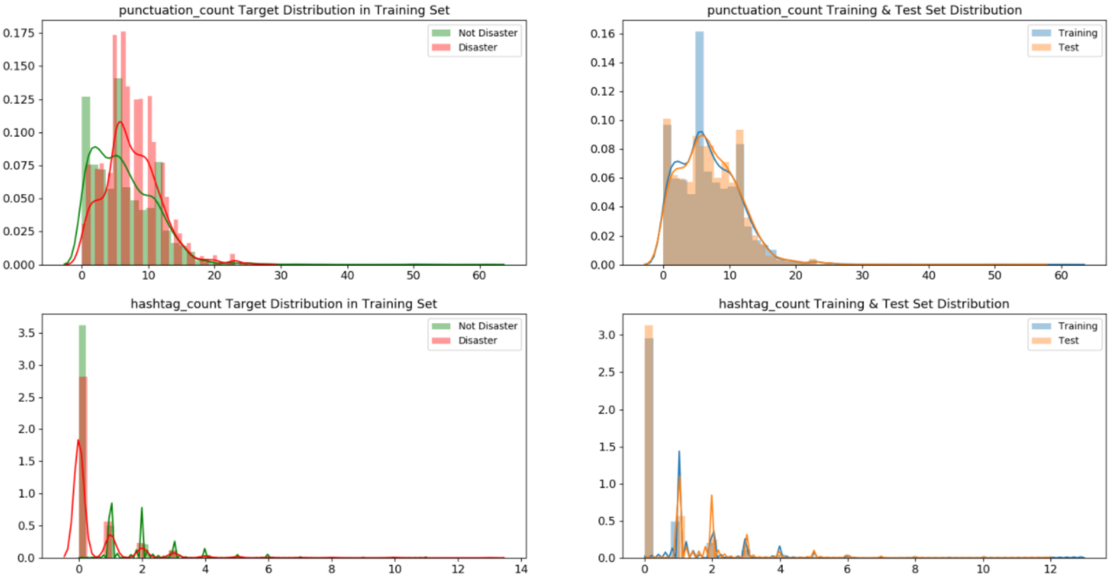







stop_word_count Target Distribution in Training Set
Not Disaster
Disaster
0.00
0.05
0.10
0.15
0.20
0.25
0
5
10
15
20
stop_word_count Training & Test Set Distribution
Training
Test
0.00
0.05
0.10
0.15
0.20
0.25
0
5
10
15
20
url_count Target Distribution in Training Set
Not Disaster
Disaster
0
1
2
3
4
5
0
1
2
3
4
url_count Training & Test Set Distribution
Training
Test
0
1
2
3
4
5
0
1
2
3
4


mean_word_length Target Distribution in Training Set
Not Disaster
Disaster
0.00
0.05
0.10
0.15
0.20
0.25
0.30
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
mean_word_length Training & Test Set Distribution
Training
Test
0.00
0.05
0.10
0.15
0.20
0.25
0.30
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
char_count Target Distribution in Training Set
Not Disaster
Disaster
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0
25
50
75
100
125
150
175
char_count Training & Test Set Distribution
Training
Test
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
−25
0
25
50
75
100
125
150
175


punctuation_count Target Distribution in Training Set
Not Disaster
Disaster
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0
10
20
30
40
50
60
punctuation_count Training & Test Set Distribution
Training
Test
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0
10
20
30
40
50
60
hashtag_count Target Distribution in Training Set
Not Disaster
Disaster
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
2
4
6
8
10
12
14
hashtag_count Training & Test Set Distribution
Training
Test
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
2
4
6
8
10
12

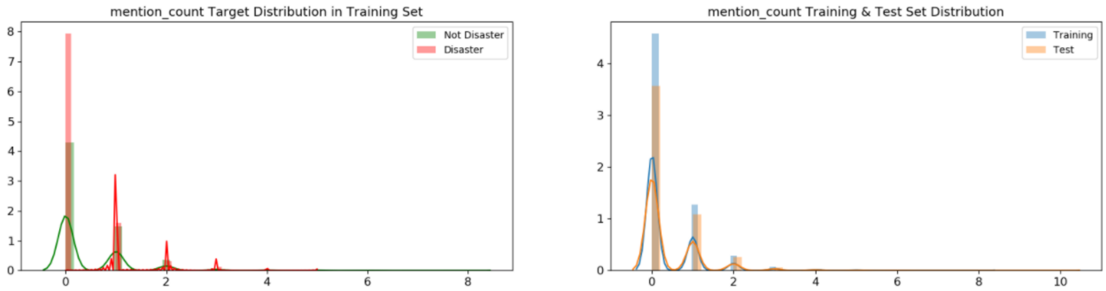


Figure 5.7 Analysis of meta features.

5.5 N-Grams

5.5.1 Unigrams

The most common unigrams are stop words, punctuations, and numbers which indicates us that it is better to clean them before modeling as they do not provide much information about the target.

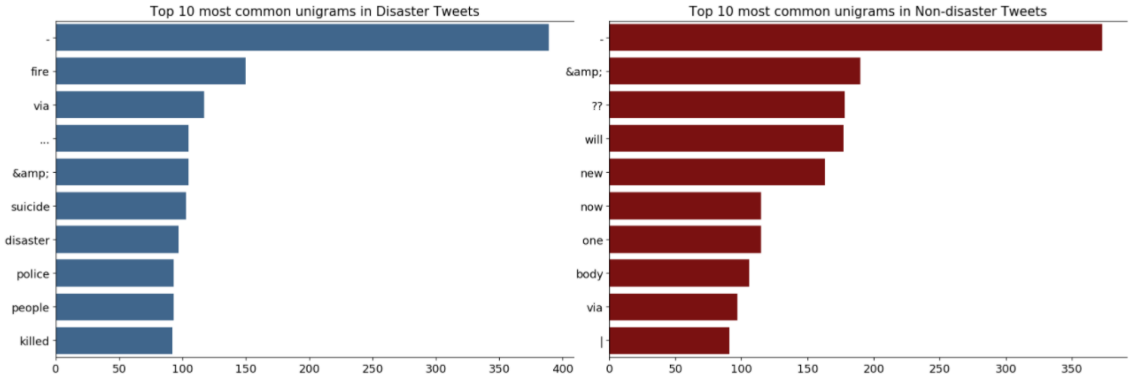


Figure 5.8 Top 10 most common unigrams in Tweets.

5.5.2 Bigrams

Clarity which refer to situation after data cleaning of the context makes the most common bigrams and trigrams losing their generality. The most common bigrams provides more information about disasters comparing with the unigrams. However, punctuation should be removed from words. The most common bigrams in non-disaster has lots of punctuations due to its characteristic which mostly comes from youtube or reddit texts. These punctuation marks should be cleared out of words.

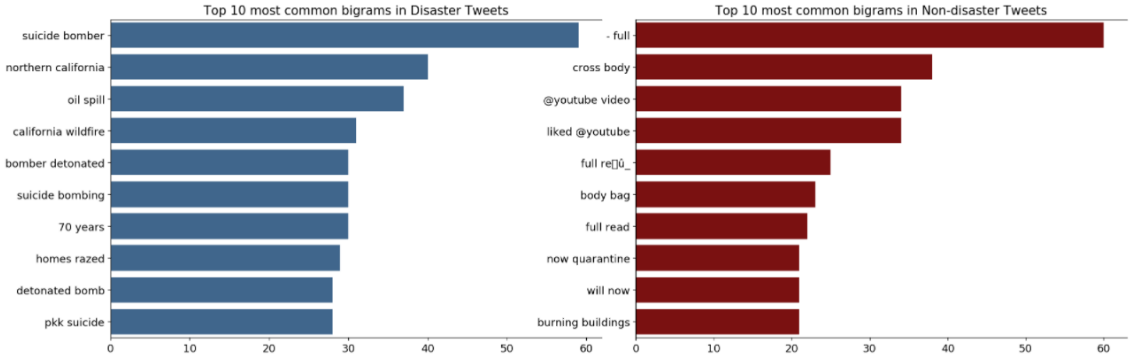


Figure 5.9 Top 10 most common bigrams in Tweets.

5.5.3 Trigrams

The most common trigrams in disaster tweets are quite similar to bigrams. Trigrams include lots of information about disasters. However, trigrams might not be useful more than bigrams, because, may mostly not give any additional information. In non-disaster tweets, trigrams contain even more punctuations than bigrams.

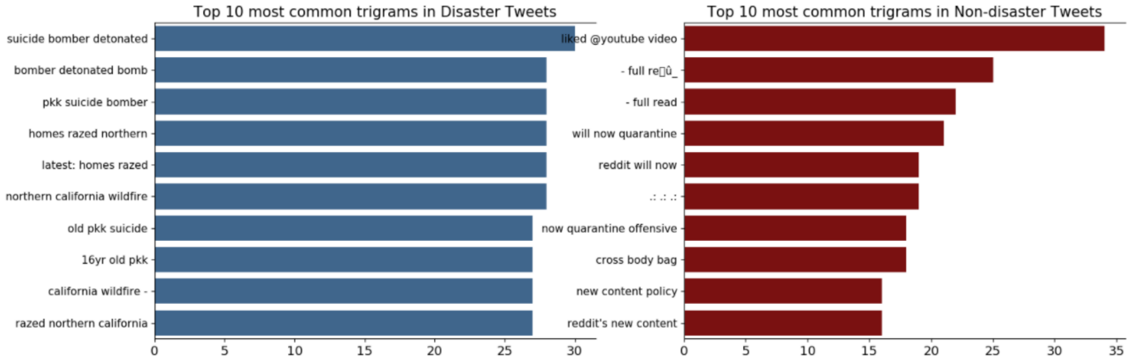


Figure 5.10 Top 10 most common trigrams in Tweets.

5.6 Data Cleaning

There are noises that need to be cleaned up in the raw data from Twitter. A general approach will be used for cleaning. Thus, system applies a pre-processing phase to remove hashtags, punctuation marks, and URLs if necessary. Exploratory data analysis above shows that tweet texts need to be cleaned before using as training data. Meanwhile, stop-words are removed from the sentence quite often since take up appreciable space and their removal does not make any significant difference. For cleaning process we observed different situations for BERT and GloVe, and will be explained in 8.

5.7 The Explanation of Different Proposed Learning Pipelines

Figure 5.11 presents the proposed GloVe-CNN-BiLSTM model pipeline, which includes of three consecutive modules. GloVe produces competitive results. Besides that, 5.12 presents the proposed BERT-CNN-BiLSTM model pipeline, which also includes three consecutive modules. BERT shows its strength in our task, and can be considered as the new electricity of natural language processing tasks such as sentiment analysis, named entity recognition, and topic modeling. A particular combination design is required to be able to use strengths of both CNN and BiLSTM models:

- In text processing, 1D CNN layers are known for their capability to extract as many features as possible.
- BiLSTM is keeping the chronological order between words in given texts, meanwhile, it can ignore the unnecessary words by utilizing the delete gate.

Combining CNN and BiLSTM has a particular purpose. We want to create a model pipeline that leverages the strengths of both to capture features extracted using CNN and use them as BiLSTM inputs. Thus, we proposed a model that meets this goal. Word embedding part which includes BERT and GloVe is used as the input of convolutional neural network. Then takes features as input of BiLSTM layers. Afterwards, the output of the BiLSTM is fed to a detection head to produce the final results, in other saying, disaster or not disaster.

5.7.1 GloVe-CNN-BiLSTM

GloVe provides the ability to derive semantic relationships between words from co-occurrence matrices. Given a corpus having V words, the co-occurrence matrix X will be a V x V matrix, where the i^{th} row and j^{th} column of X, X_{ij} denotes how many times word i has co-occurred with word j. GloVe gives rest of the model a statistical word embeddings to utilize dataset.

5.7.2 BERT-CNN-BiLSTM

BERT is used to transform word tokens from text data to contextual word embeddings. BERT produces competitive prediction results against the models like GloVe. This is because every word under GloVe has a fixed expression regardless of the context in which the word appears, however, BERT produces a word expression that is dynamically influenced by the surrounding words. This situation creates a difference in pre-processing step, and will be explained in section 8.

Figure 5.11 GloVe-CNN-BiLSTM learning pipeline with extended visualization.

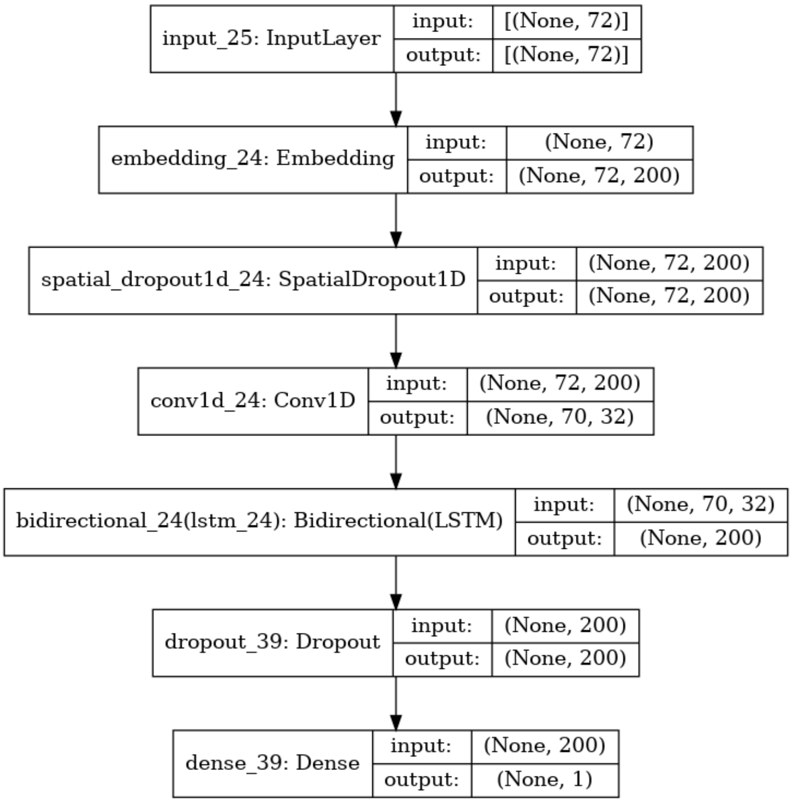
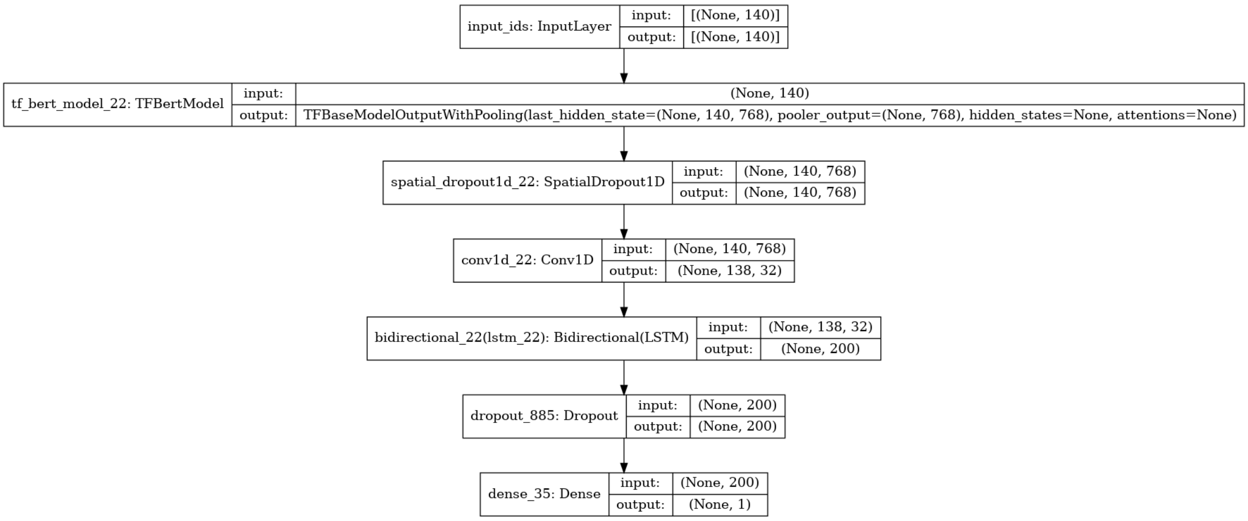


Figure 5.12 BERT-CNN-BiLSTM learning pipeline with extended visualization.



6
Implementation

The dataset was separated into training, validation and test sets with the ratio of (9:1):1, we validate our model with, 6165 training, 762 validation, and 686 test samples. For GloVe embedding dimension was 200 and learning rate was 1e-4 *glove.twitter.27B.200d.txt* is used as base embeddings. For BERT embeddings "*bert-base-cased*" is used which includes 12-layer, 768-hidden, 12-heads. For CNN module 32 dimensions are used with 3 filter size. BiLSTM is set to 100 layers with 0.2 dropout on both way. For the overall architecture, learning rate 1e-4, Adam optimizer is used, and experimented with batch size 32 and training epoch 3. All experiments were realized with Python 3.9.7 and Tensorflow 2.7.0 on Kaggle Notebook with TPU v3-8.

7
Performance Analysis

For performance evaluation we use the disaster Tweet dataset discussed in Section 5.4. First, we discuss about the performance metrics, afterward, report of the experimental results. We will present an analysis of errors and performances of different pipelines that might give intuition on further improvement of the model.

7.1 Evaluation Metrics

Precision, Recall, and F1 scores are used to evaluate the performance of models. Positive and negative samples are unbalanced. Thus, F1 score gives better results in this case than accuracy. Precision and recall are also significant as evaluation metrics. The first one represents the number of false positives. If the accuracy increases, the false positives decreases.

The second shows the number of wrong predicted positive samples. If the recall increases, the missed disaster tweets decreases. Meanwhile, precision and recall gap should be small. Hence, we can consider single metric is not directing the model. In our case, F1 score needs to be optimized. Let TP, TN, and FP represent the number of true positives, true negatives, and false positives, in order, and we can compute precision in equation (7.1), recall in equation (7.2), and F1 in equation (7.3).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7.1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7.2}$$

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7.3}$$

8
Experimental Results

We performed experiments to evaluate a set of modules, and presented as performance comparison of all evaluated modules in the table 8.1. The set of models CNN, BiLSTM, and CNN-BiLSTM forms an ablation study with GloVe and BERT embeddings, thus, we can evaluate the many situational effect case on the performance such as existence of pre-processing, and type of embeddings for each individual model and the combinations. We can see that the single-layer CNN model performs unsatisfying results with GloVe embeddings and cannot learn any contextual information, however, a single-layer CNN can learn more contextual information using BERT embeddings. BERT presents an obvious improvement as against GloVe embeddings. BERT utilizes raw dataset better than GloVe embeddings, because as shown in [21] BERT is sensitive to punctuations and cases while creating contextual matrix. On the other hand, GloVe requires data cleaning, because stopwords, punctuations etc. drive context to statistically wrong way. We can easily see the difference on models along with (AP) After Pre-processing in performance table 8.1. BERT loses its contextual structure which depends on the punctuations and cases while GloVe gains remarkable score when we eliminate the unnecessary punctuations and cases. Overall, CNN-BiLSTM shows its strength in results. Our final pipeline, BERT-CNN-BiLSTM is above all other models, and shows the power of combining the strength of each building block.

Table 8.1 A performance comparison of models on raw dataset. If pre-processed dataset is used, determined as *(AP) After Pre-process*.

Model	F1 Score	Precision	Recall
GloVe-CNN	63.49	71.65	57.08
GloVe-BiLSTM	73.97	86.19	65.06
BERT-CNN-BiLSTM *(AP)*	77.32	85.28	70.92
BERT-BiLSTM	77.44	**87.61**	69.49
BERT-CNN	77.63	86.99	70.17
GloVe-CNN-BiLSTM	78.13	79.22	77.73
GloVe-CNN-BiLSTM *(AP)*	78.71	78.83	**79.10**
BERT-CNN-BiLSTM	**79.10**	84.74	74.23

8.1 Error Analysis of BERT-CNN-BiLSTM Pipeline

Table 8.2 presents ten samples. It contains five positive and five negative samples, which are misclassified by the BERT-CNN-BiLSTM model. The first five samples (1-5) are marked as disaster, and for the next five samples (6-10), none of them represent a common sense disaster.

8.1.1 Mislabeled Texts in Raw Data

We observed that there are some mislabeled samples in the dataset. This situation affects performances of the models. Sample 1 clearly mislabeled in dataset, it is a general sentence which talks about evil of people from perspective of the user, but we cannot classify this sentence as disaster. In sample 5, sentence is clearly mislabeled, general vibe of the sentence with following word "thruuu" is daily issue of the user.

8.1.2 Word Ambiguity

Word ambiguity is one of the important problem when it comes to open ideas of people in social media. Therefore, we observed that word ambiguity is an actual problem while classifying. Sample 2 has word ambiguity, everything related with the disaster such as "Breaking News!", "loud bang", and "blast of wind" so it can be disaster, but there might a chance the user tries to make a joke to mess with his/her neighbour.

8.1.3 Actual Misclassified Texts by Proposed Model

To improve model we need to understand why the model misclassified the proper texts. Therefore, we cover the actual misclassified texts in the following cases. Sample 3 is an actual alert; sample 4 includes phase "mass murder" and obvious explanation of violence. In sample 6 the word "lmfao" almost every time used in entertaining moments; sample 7 talks about courage of user's friend; sample 8 is quite close to an actual disaster, however, the word "simulate" changes the semantic meaning to its reverse; sample 9 includes phase "burning buildings", but the rest of sentence irrelevant with the any disaster; sample 10 has bad meaning religiously, but cannot considered as disaster.

Table 8.2 Missclassified examples. A "+" notation represents a disaster sample, and a "-" notation represents a not disaster sample.

ID	Text	Label	Predicted
1	Some people are really natural disaster too	+	-
2	Breaking news! Unconfirmed! I just heard a loud bang nearby. in what appears to be a blast of wind from my neighbour's ass.	+	-
3	ALERT! Sandy Hook Elementary School Evacuated After Bomb Threat	+	-
4	We have different moral systems. Mine rejects the mass murder of innocents yours explicitly endorses such behavior.	+	-
5	So much shit has happened today wtf idk how I survive thruuu it all	+	-
6	Our garbage truck really caught on fire lmfao.	-	+
7	My man runs into burning buildings for a living but is scared to hit up a girl. I don't get it.	-	+
8	Google Alert: Emergency units simulate a chemical explosion at NU	-	+
9	?? your last retweet you would think the lion saved people from a burning buildings it's not that deep	-	+
10	China detains seven Christians trying to protect their church's cross from demolition	-	+

These samples are randomly chosen from all error predictions as misclassifed examples which can be seen in the table. The tweet length limit has pros and cons to train a disaster tweet classifier. The good thing is that every user has to write short, meaningful words to explain their ideas due to character restriction by Twitter. The negative side is that many short Tweets just mean nothing. It is quite difficult to give a meaning due to lack of additional contextual information. This is one of the most significant problem with building an accurate model pipeline from Twitter data.

9
Conclusion

Classifying disaster tweets is closely related to people's daily lives, and research efforts in this field have increased in recent years. Disaster prediction studies help raise awareness, improve state rescue mechanisms, and plan charitable activities. This study analyzes a new model pipeline for classifying disaster tweets. Our last model pipeline, BERT-CNN-BiLSTM, uses a BERT encoder, a 1D convolutional, and a BiLSTM layer to extract high quality linguistic features for classifying the disaster tweets from the regular ones. This model pipeline was validated against competitors through extensive experimentation which makes it a promising model pipeline for applying to real time disaster detection systems. Although the proposed model pipeline was trained and validated on an English dataset, it can be applied to datasets in other languages.

References

[1] K. Zahra, M. Imran, and F. O. Ostermann, "Automatic identification of eyewitness messages on twitter during disasters," *& Management*, vol. 57, no. 1, p. 102 107, 2020, ISSN: 0306-4573. DOI: `https://doi.org/10.1016/j.ipm.2019.102107`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0306457319303590`.

[2] M. Imran, P. Mitra, and C. Castillo, "Twitter as a lifeline: Human-annotated twitter corpora for NLP of crisis-related messages," *CoRR*, vol. abs/1605.05894, 2016. arXiv: `1605.05894`. [Online]. Available: `http://arxiv.org/abs/1605.05894`.

[3] R. R. Arinta and E. Andi W.R., "Natural disaster application on big data and machine learning: A review," in *2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, 2019, pp. 249–254. DOI: `10.1109/ICITISEE48480.2019.9003984`.

[4] H. Shekhar and S. Gangisetty, "Disaster analysis through tweets," Aug. 2015. DOI: `10.1109/ICACCI.2015.7275861`.

[5] J. R. Chowdhury, C. Caragea, and D. Caragea, "Keyphrase extraction from disaster-related tweets," *CoRR*, vol. abs/1910.07897, 2019. arXiv: `1910.07897`. [Online]. Available: `http://arxiv.org/abs/1910.07897`.

[6] M. Imran, S. Elbassuoni, C. Castillo, F. Diaz, and P. Meier, "Practical extraction of disaster-relevant information from social media," May 2013. DOI: `10.1145/2487788.2488109`.

[7] S. Zong, A. Baheti, W. Xu, and A. Ritter, "Extracting COVID-19 events from twitter," *CoRR*, vol. abs/2006.02567, 2020. arXiv: `2006.02567`. [Online]. Available: `https://arxiv.org/abs/2006.02567`.

[8] G. Song and D. Huang, "A sentiment-aware contextual model for real-time disaster prediction using twitter data," *Future Internet*, vol. 13, no. 7, 2021, ISSN: 1999-5903. [Online]. Available: `https://www.mdpi.com/1999-5903/13/7/163`.

[9] A. Kumar, J. P. Singh, and S. Saumya, "A comparative analysis of machine learning techniques for disaster-related tweet classification," in *2019 IEEE R10 Humanitarian Technology Conference (R10-HTC)(47129)*, 2019, pp. 222–227. DOI: `10.1109/R10-HTC47129.2019.9042443`.

[10] M. Y. Kabir and S. Madria, "A deep learning approach for tweet classification and rescue scheduling for effective disaster management," *CoRR*, vol. abs/1908.01456, 2019. arXiv: `1908.01456`. [Online]. Available: `http://arxiv.org/abs/1908.01456`.

[11] A. Khatri, P. P, and A. K. M, “Sarcasm detection in tweets with BERT and glove embeddings,” *CoRR*, vol. abs/2006.11512, 2020. arXiv: 2006.11512. [Online]. Available: https://arxiv.org/abs/2006.11512.

[12] S. Madichetty and S. Muthukumarasamy, “Detection of situational information from twitter during disaster using deep learning models,” *Sādhanā*, vol. 45, Dec. 2020. DOI: 10.1007/s12046-020-01504-0.

[13] B. Jang, M. Kim, G. Harerimana, S.-u. Kang, and J. W. Kim, “Bi-lstm model to increase accuracy in text classification: Combining word2vec cnn and attention mechanism,” *Applied Sciences*, vol. 10, no. 17, 2020, ISSN: 2076-3417. DOI: 10.3390/app10175841. [Online]. Available: https://www.mdpi.com/2076-3417/10/17/5841.

[14] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, and L. He, “A survey on text classification: From shallow to deep learning,” *CoRR*, vol. abs/2008.00364, 2020. arXiv: 2008.00364. [Online]. Available: https://arxiv.org/abs/2008.00364.

[15] J. Pennington, R. Socher, and C. Manning, *Glove: Global vectors for word representation*, Jan. 2014. DOI: 10.3115/v1/D14-1162.

[16] C. Liu, P. Zhang, T. Li, and Y. Yan, “Semantic features based n-best rescoring methods for automatic speech recognition,” *Applied Sciences*, vol. 9, p. 5053, Nov. 2019. DOI: 10.3390/app9235053.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *CoRR*, vol. abs/1706.03762, 2017. arXiv: 1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[19] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[20] Y. Kim, “Convolutional neural networks for sentence classification,” *CoRR*, vol. abs/1408.5882, 2014. arXiv: 1408.5882. [Online]. Available: http://arxiv.org/abs/1408.5882.

[21] A. Ek, J.-P. Bernardy, and S. Chatzikyriakidis, “How does punctuation affect neural models in natural language inference,” in *Proceedings of the Probability and Meaning Conference (PaM 2020)*, Gothenburg: Association for Computational Linguistics, Jun. 2020, pp. 109–116. [Online]. Available: https://aclanthology.org/2020.pam-1.15.

Curriculum Vitae

FIRST MEMBER

Name-Surname: Toygar Tanyel
Birthdate and Place of Birth: 14.04.2000, İzmir
E-mail: l1118094@std.yildiz.edu.tr
Phone: 546 416 2000
Practical Training: AI Research Intern
Project: Sort Optimization of Training Samples in Machine Learning
Advisor: Assoc. Prof. Mehmet Fatih AMASYALI
Dec 2021 to Present

Project System Informations

System and Software: Windows Operating System, Natural Language Processing, Machine Learning & Deep Learning, Python
Required RAM: 16GB
Required Disk: 50GB+