A Retrospective Study of Disaster Events for Designing a Disaster Intent Spatiotemporal Query Set.

SURAJ

A Retrospective Study of Disaster Events for Designing a Disaster Intent Spatiotemporal Query Set.

Project-I Report submitted in partial fulfillment for the award of degree of

Master of Computer Applications

in

Computer Science & Engineering

by

Suraj

18CS6002

Under the supervision of

Dr. Sujoy Saha

DEPARTMENT OF ELECTRICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY DURGAPUR

Declaration

I certify that

1. The work contained in this thesis is original and has been done by me under the guidance of my supervisor.
2. The work has not been submitted to any other Institute for any degree or diploma.
3. I have followed the guidelines provided by the Institute in preparing the thesis.
4. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
5. Whenever I have used materials (data, theoretical analysis, figures and text) from other sources, I have given due credit to them by citing them the text of the thesis and giving their details in the references.

SURAJ

Certificate Of Approval

This is to certify that the thesis entitled **"A Retrospective Study of Disaster Events for Designing a Disaster Intent Spatiotemporal Query Set"**, submitted by **SURAJ** for the partial fulfillment of the requirements for the award of the degree of **Master of Computer Applications**, is a bonafide research work under the guidance of **Dr. Sujoy Saha**. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma. In our opinion, this thesis is of the standard required for the partial fulfilment of the requirements for the award of the degree of **Master of Computer Applications**.

(Counter Signed by)
Dr. Sujoy Saha
Asst. Professor, Dept. of CSE
National Institute of Technology
Durgapur-713209, INDIA

(Counter Signed by)
Prof. Tandra Paul
Prof. & Head, Dept. of CSE
National Institute of Technology
Durgapur-713209, INDIA

Acknowledgements

First and foremost, I thank my research supervisors **Dr. Sujoy Saha**. Without their assistance and dedicated involvement in every step throughout the process, this thesis would never have been accomplished.
I wish to acknowledge **Dr. Tandra Paul, HOD of CSE**, for providing a supporting environment for my work. I would also like to show gratitude and respect to other faculty members of our department for their encouragement.
I thank **Hemant Sarraf**, my project fellow for the project discussions, for helping in need and for all the fun we had in the last 6 months.
Last but not the least, I would like to thank my mother and my friends for their unconditional support.

SURAJ

Abstract

This study focuses on designing of temporal query set through analyzing the community sensing in Social Media i.e. Twitter in different disaster events. Community sensing in social media i.e. Twitter is very significant nowadays as it provides the perception of any human being about any type of events taking place all over the world. Such conception is also true for any disaster event for which millions of tweets can be detected based on various situational as well as non-situational aspects. Through spatiotemporal analysis of such tweets, the flow of different types of disaster intent information at different times can be detected for different disaster cases. Such type of information, once analyzed can be made effective in designing proper query sets for any conversational application, trying to follow proper data acquisition rules in an online or offline environment. In current work, we mainly focus on designing a question bank for different disaster intent information classes which are required for community sensing applications both in offline/online environments in order to generate a systematic information flow for effective situational assessment and response in disaster scenarios. Here, we study the community sensing from past disaster data sets (floods, cyclones, earthquakes and landslides) posted in social media i.e. Twitter. Here, Firstly, we classify the tweets based on defining distinct features considering the notion of the information classes.

Secondly, after classification, we study and analyze the spatial and time varying propagation of those information classes for various disaster data sets. Finally, after finding time varying correlation among the propagation of the information classes, we then design questionnaires for each of the information classes which can be used for information acquisition using any offline/online crowdsensing applications.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Communication during and immediately after a disaster situation is an important component of response and recovery, in that it connects affected people, families, and communities with first responders, support systems, and other family members. Reliable and accessible communication and information systems also are key to a community's resilience.

1.1 Background Study

Our present system, **Surakshit** is an android application for connecting people and gathering important information after disasters. Internet and access routes to disaster areas play an elementary role in disaster response and recovery. These help in faster information transmission and easier reach to victims by volunteers and other helping bodies. Our application makes use of the internet if network links are available, else it establishes a local network among people in the surroundings using wifi **P2P**. The information gathered using the latter is collected by a volunteer and thereby passed on to authorities and responders.

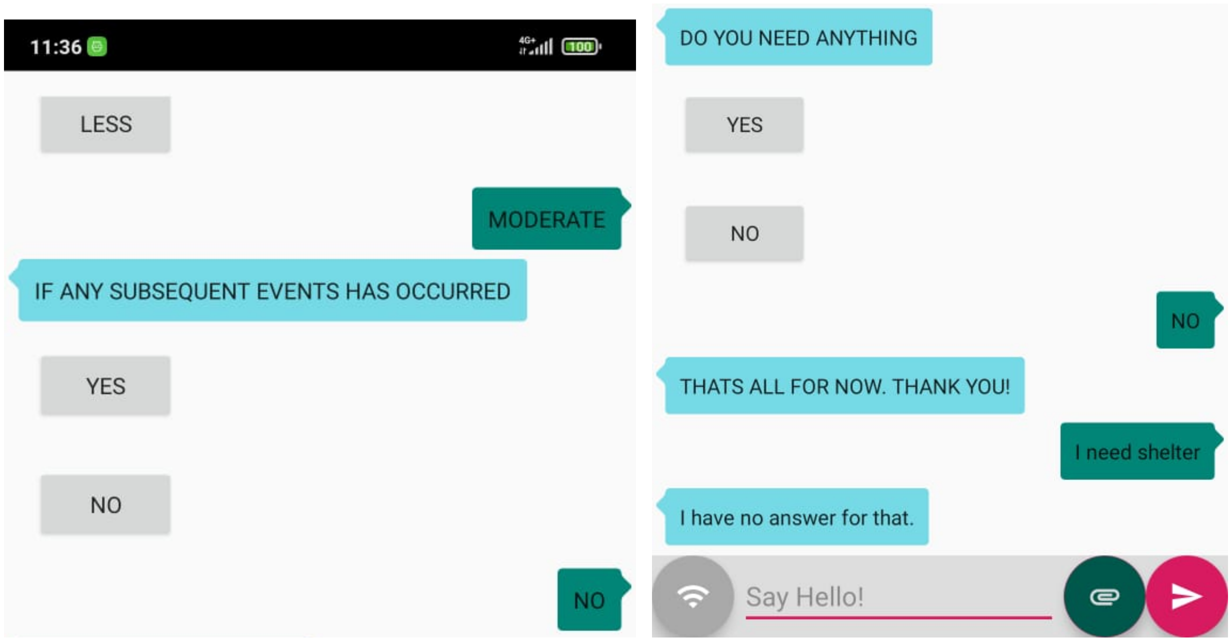
Chatbot is a software application used for chat conversation with users impersonating human behaviours. These help the users to resolve common problems. In our application, we are using it to collect information from users. The chatbot interacts with users with an efficient dialog flow, collects and provides all necessary information. It serves as a convenient option for users to interact with our application. The current dialog flow of our chatbot has been implemented using AIML (Artificial Intelligence Markup Language).

1.1.1 AIML Chatbot

AIML is a form of XML that defines rules for matching patterns and determining responses. With AIML it is possible to create human interfaces, keeping the implementation simple to program, easy to understand and highly maintainable. We prepared a set of questions for each group of people. The dialog flow is controlled by the responses.

Although it was good enough for simple chat flow, the diverse real life scenarios needed the chat bot to be intelligent. Any set of templates would not be enough to make the bot smart. It should be intelligent enough for abstract thinking, reasoning and creation.

The query set was an abstract preparation based on basic needs and queries of people in disaster events. The actual query set required needs to be vast enough for all groups of people. Also, preparing such complex templates to handle all groups of people with their diverse scenarios is very difficult.


11:36

LESS

MODERATE

IF ANY SUBSEQUENT EVENTS HAS OCCURRED

YES

NO

NO

DO YOU NEED ANYTHING

YES

NO

NO

THATS ALL FOR NOW. THANK YOU!

I need shelter

I have no answer for that.

Figure 1.1: AIML Figures

As shown in the Fig., the chatbot is able to answer simple, pattern matching queries but fails when the query is out of context.

1.2 Problem Statement

To address the problems faced in our previous implementation, we need to prepare a query set based on people's sentiments during disaster scenarios. This query set will help us to formulate the dialog flow of our chatbot.

The query set will be prepared using the tweets uploaded by users during past disaster events. The tweets will be classified based on their type of information and fed to a machine learning model for training. The data from tweets will help our chatbot understand diverse real life scenarios.

Chapter 2
Related Work

2.1 LSTM Neural Networks

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition and anomaly detection in network traffic or IDSs (intrusion detection systems).

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series.

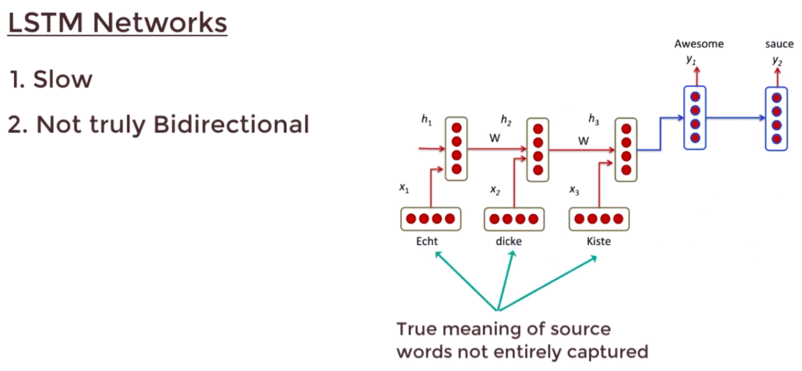


Fig. 2.1 LSTM

2.1.1 Limitation of LSTM Networks

As it is said, everything in this world comes with its own advantages and disadvantages, LSTMs too, have a few drawbacks which are discussed as below:

1. LSTMs became popular because they could solve the problem of vanishing gradients. But it turns out, they fail to remove it completely.

2. They require a lot of resources and time to get trained and become ready for real-world applications. In technical terms, they need high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide for. Thus, hardware-wise, LSTMs become quite inefficient.
3. LSTMs mostly work with methods of Unidirectional parsing and hence behave quite similar to that of a feed-forward neural net.
4. LSTMs are prone to overfitting and overfitting is not a good curve to train your data.

This gives us the idea of moving from an inefficient model to a better model which is a Transformer.

2.2 Transformer Network

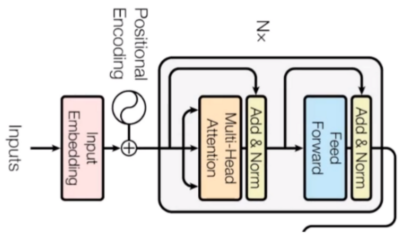
A paper called "Attention Is All You Need" published in 2017 comes into the picture, it introduces an encoder decoder architecture based on attention layers, termed as the transformer.

One main difference is that the input sequence can be passed parallelly so that GPU can be utilized effectively, and the speed of training can also be increased. And it is based on the multi-headed attention layer, vanishing gradient issue is also overcome by a large margin. The paper is based on the application of transformer on NMT(Neural Machine Translation). So, here both of our problems which we highlighted before are solved to some level here.

Like for example in a translator made up of simple RNN we input our sequence or the sentence in a continuous manner, one word at a time to generate word embeddings. As every word depends on the previous word, it's hidden state acts accordingly, so it is necessary to give one step at a time. While in a transformer, it is not like that, we can pass all the words of a sentence simultaneously and determine the word embedding simultaneously.

2.2.1 Architecture

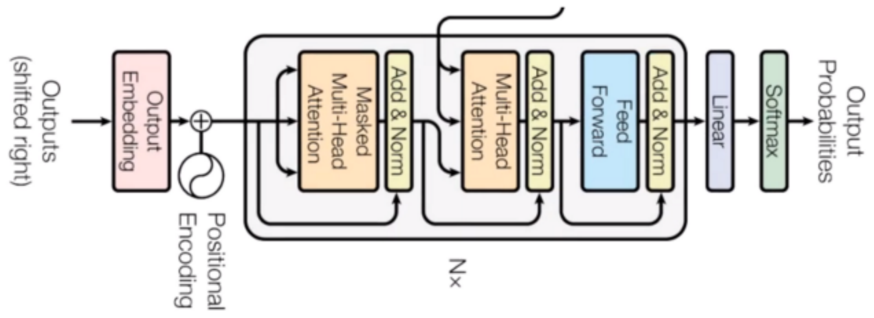
1. Encoder Block



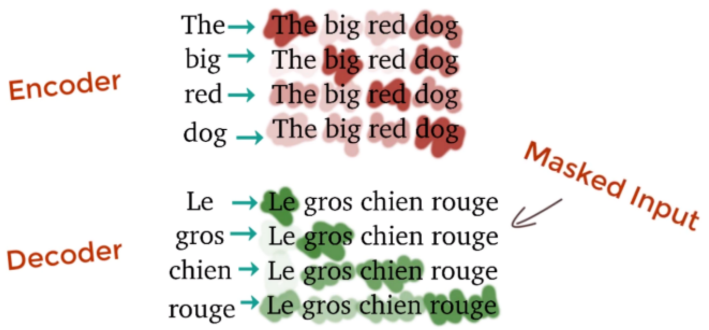
It's a fact that computers don't understand words, it works on numbers, vectors or matrices. So, we do need to convert our words to a vector. But how can this be possible? So, here the concept of Embedding Space comes. It's like an open space or dictionary where words of similar meanings are grouped together or are present close to each other in that space.

This space is termed as embedding space, and here every word, according to its meaning, is mapped and assigned with a particular value. So, here we convert our words to vectors.
Word → **Embedding** → **Positional Embedding** → **Final Vector**, termed as **Context**.

2. Decoder Block



Now, like if we are training a translator for English to the French language, so for training we need to give an English sentence along with it's translated French sentence for the model to learn. So, our English sentences pass through the Encoder Block, and French sentences pass through the Decoder Block.



So, this is how the transformer works, and it is now the state-of-the-art technique in NLP. It is giving wonderful results, using a self-attention mechanism and also solves the parallelization issue. Even Google uses BERT that uses a transformer to pre-train models for common NLP applications.

2.3 Why BERT?

Bidirectional Encoder Representations from Transformers is a Transformer-based machine learning technique for natural language processing pre-training developed by Google. It is a method of pre-training language representations, meaning that we train a general-purpose ”language understanding” model on a large text corpus (like Wikipedia), and then use that model for downstream **NLP** tasks that we care about (like question answering).

BERT outperforms previous methods because it is the first unsupervised, deeply bidirectional system for pre-training **NLP**.

Unsupervised means that **BERT** was trained using only a plain text corpus, which is important because an enormous amount of plain text data is publicly available on the web in many languages.

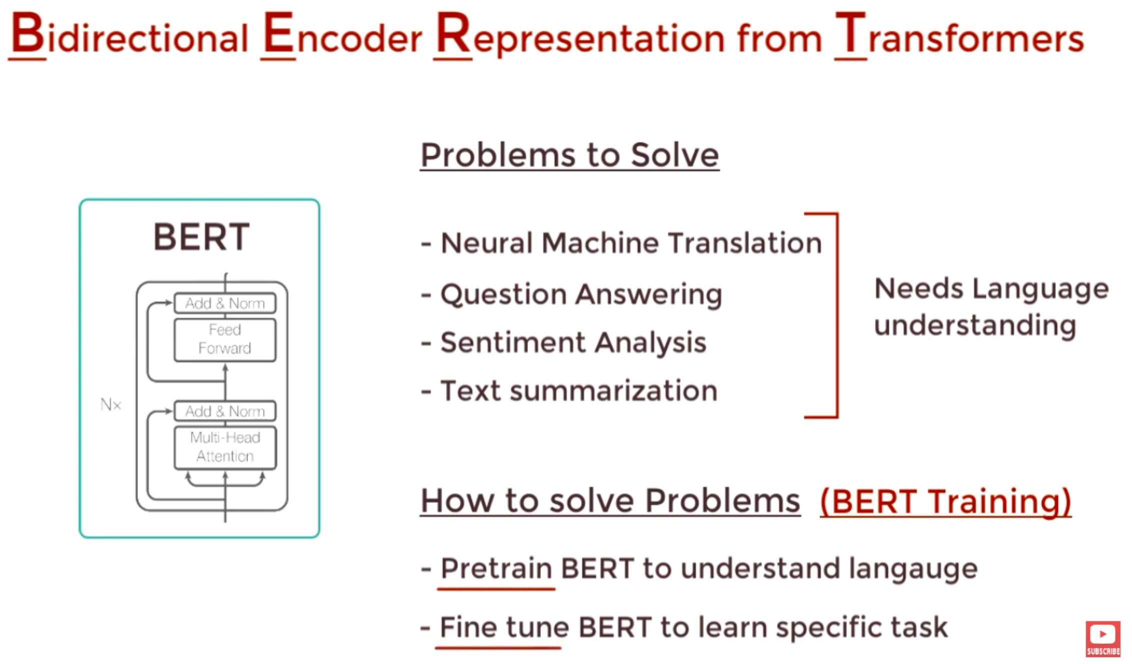
Pre-trained representations can also either be context-free or contextual, and contextual representations can further be unidirectional or bidirectional.

2.3.1 How BERT works?

BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT’s goal is to generate a language model, only the encoder mechanism is necessary.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional, though it would be more accurate to say that it’s non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

2.3.2 How to use BERT? (Fine - tuning)


Bidirectional Encoder Representation from Transformers

Problems to Solve

- Neural Machine Translation
- Question Answering
- Sentiment Analysis
- Text summarization

Needs Language understanding

How to solve Problems (BERT Training)

- Pretrain BERT to understand langauge
- Fine tune BERT to learn specific task

BERT
Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head Attention
SUBSCRIBE

Fig. 2.1 BERT

Using BERT for a specific task is relatively straightforward:
BERT can be used for a wide variety of language tasks, while only adding a small layer to the core model:

1. Classification tasks such as sentiment analysis are done similarly to Next Sentence classification, by adding a classification layer on top of the Transformer output for the [CLS] token.
2. In Question Answering tasks (e.g. SQuAD v1.1), the software receives a question regarding a text sequence and is required to mark the answer in the sequence. Using BERT, a Q&A model can be trained by learning two extra vectors that mark the beginning and the end of the answer.

Chapter 3
Data Acquisition

3.1 Platform For Data

Twitter

Twitter is an American microblogging and social networking service on which users post and interact with messages known as ”tweets”. Registered users can post, like and retweet tweets, but unregistered users can only read them. Users access Twitter through its website interface, through Short Message Service (SMS) or its mobile-device application software (”app”). Twitter, Inc. is based in San Francisco, California, and has more than 25 offices around the world.[14] Tweets were originally restricted to 140 characters, but was doubled to 280 for non-CJK languages in November 2017.Audio and video tweets remain limited to 140 seconds for most accounts. By 2012, more than 100 million users posted 340 million tweets a day, and the service handled an average of 1.6 billion search queries per day. In 2013, it was one of the ten most-visited websites and has been described as ”the SMS of the Internet”.As of 2018, Twitter had more than 321 million monthly active users.

Tweets

Tweets are publicly visible by default, but senders can restrict message delivery to only their followers. Users can mute users they do not wish to interact with and block accounts from viewing their tweets.Users can tweet via the Twitter website, compatible external applications (such as for smartphones), or by Short Message Service (SMS) available in certain countries. Users may subscribe to other users' tweets—this is known as ”following” and subscribers are known as ”followers”or ”tweeps”, a portmanteau of Twitter and peeps.[136] Individual tweets can be forwarded by other users to their own feed, a process known as a ”retweet”. Users can also ”like” (formerly ”favorite”) individual tweets.

Twitter allows users to update their profile via their mobile phone either by text messaging or by apps released for certain smartphones and tablets. Twitter has been compared to a web-based Internet Relay Chat (IRC) client. In a 2009 Time magazine essay, technology author Steven Johnson described the basic mechanics of Twitter as ”remarkably simple”:

3.2 Tools for Data Collection

3.3 TWINT (open source Python Module)

Twint is an advanced Twitter scraping tool written in Python that allows for scraping Tweets from Twitter profiles without using Twitter's API.Twint utilizes Twitter's search operators to let you scrape Tweets from specific users,scrape Tweets relating to certain topics, hashtags trends, or sort out sensitive information from Tweets like email and phone numbers. I find this very useful, and you can get really creative with it too.Twint also makes special queries to Twitter allowing you to also scrape a Twitter user's followers, Tweets a user has liked, and who they follow without any authentication, API, Selenium, or browser emulation.

3.3.1 Installing & Using TWINT

GIT :

1. git clone –depth=1 https://github.com/twintproject/twint.git
2. cd twint
3. pip3 install . -r requirements.txt

PIP :

1. pip3 install twint

PIPenv :

1. pipenv install git+https://github.com/twintproject/twint.gitegg=twint

A few simple examples to help you understand the basics:

- `twint -u username` - Scrape all the Tweets from the user's timeline.
- `twint -u username -s pineapple` - Scrape all Tweets from the *user*'s timeline containing *pineapple*.
- `twint -s pineapple` - Collect every Tweet containing *pineapple* from everyone's Tweets.
- `twint -u username --since 2015-12-20` - Collect Tweets that were tweeted since 2015-12-20 00:00:00.
- `twint -u username -o file.csv --csv` - Scrape Tweets and save as a csv file.
- `twint -u username --email --phone` - Show Tweets that might have phone numbers or email addresses.
- `twint -s "Donald Trump" --verified` - Display Tweets by verified users that Tweeted about Donald Trump.
- `twint -u username -o file.json --json` - Scrape Tweets and save as a json file.

3.3.2 Limitations of TWINT

All of public Twitter is still available through its standard web interface. The Python library twint takes advantage of this so you can collect data from Twitter without using the API. While it's pretty powerful, one major limitation is that while it gives the count of times something has been liked or retweeted, it does not return who liked or retweeted it.

Also it was an unofficial way to collect data from twitter so we have to give up this module because we cannot simply use any unauthorised tools for collecting tweets as data. Also TWINT limits scrolls while browsing the user timeline. This means that with .Profile or with .Favorites you will be able to get 3200 tweets only.

3.4 TWEEPY (open source Python Module)

Python is a great language for all sorts of things. Very active developer community creates many libraries which extend the language and make it easier to use various services. One of those libraries is tweepy. Tweepy is open-sourced, hosted on GitHub and enables Python to communicate with Twitter platform and use its API. At the time of writing, the current version of tweepy is 1.13.

It was released on January 17, and offers various bug fixes and new functionality compared to the previous version. The 2.x version is being developed but it is currently unstable so a huge majority of the users should use the regular version.

3.4.1 Installing & Using TWEEPY

Installing tweepy is easy, it can be cloned from the Github repository:

1. **git clone https://github.com/tweepy/tweepy.git**
2. **python setup.py install**

Or using easy install:

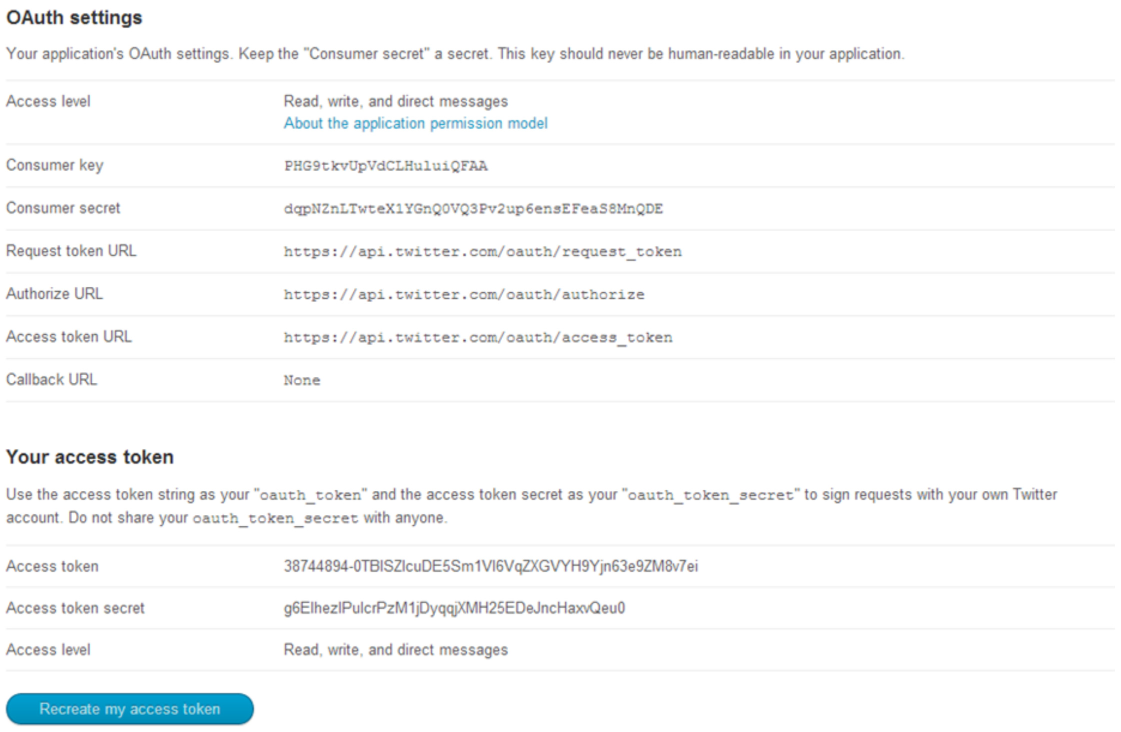
1. **pip3 install tweepy**

Using tweepy

Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API. The main difference between Basic and OAuth authentication are the consumer and access keys. With Basic Authentication, it was possible to provide a username and password and access the API, but since 2010 when Twitter started requiring OAuth, the process is a bit more complicated. An app has to be created at **dev.twitter.com**. OAuth is a bit more complicated initially than Basic Auth, since it requires more effort, but the benefits it offers are very lucrative:

- Tweets can be customized to have a string which identifies the app which was used.
- It doesn't reveal user passwords, making it more secure.
- It's easier to manage the permissions, for example a set of tokens and keys can be generated that only allows reading from the timelines, so in case someone obtains those credentials, he/she won't be able to write or send direct messages, minimizing the risk.
- The application doesn't rely on a password, so even if the user changes it, the application will still work.

After logging into the portal, and going to "Applications", a new application can be created which will provide the needed data for communicating with Twitter API.



OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read, write, and direct messages About the application permission model
Consumer key	`PHG9tkvUpVdCLHuluiQFAA`
Consumer secret	`dqpNZnLTwteX1YGnQ0VQ3Pv2up6ensEFeaS8MnQDE`
Request token URL	`https://api.twitter.com/oauth/request_token`
Authorize URL	`https://api.twitter.com/oauth/authorize`
Access token URL	`https://api.twitter.com/oauth/access_token`
Callback URL	`None`

Your access token

Use the access token string as your "`oauth_token`" and the access token secret as your "`oauth_token_secret`" to sign requests with your own Twitter account. Do not share your `oauth_token_secret` with anyone.

Access token	38744894-0TBISZlcuDE5Sm1Vl6VqZXGVYH9Yjn63e9ZM8v7ei
Access token secret	g6ElhezlPulcrPzM1jDyqqjXMH25EDeJncHaxvQeu0
Access level	Read, write, and direct messages

Fig. 3.2 Developer Account for Twitter

This is a screen which has all of the data needed to talk to the Twitter network. It is important to note that by default, the app has no access to direct messages, so by going to the settings and changing the appropriate option to "Read, write and direct messages", you can enable your app to have access to every Twitter feature.

3.4.2 Prerequisites for Data Collection

- Developer Account of Twitter
- API credentials
- Python Module
    - Tweepy
    - Csv
    - Pandas
    - sys

```
# API credentials here
consumer_key = 'BVGnYGzai5quIftplXDoyA9uC'
consumer_secret = 'aPCxlaseew1eE3Gn5dIPKeg1wAZDHJNXjEajZuVmd8nr6JcXO7'
access_token = '2900147654-f1TPy8iGWHvIEUmxPCYFJ055g0GnEqOsc1lnT8k'
access_token_secret = 'DWhKpta7V9O2AEI8wvbMg7tAB8bbjIHBmlDlVH3FV3ASE'
```

3.5 Data Volume & Format

Data Volume : Here Data is measured in terms of volume which in turns means how much amount of data is collected by the scripts. Here we have measured data volumes in term as number of tweets collected (including retweets). We have collected data from different sources manually or using automated scripts.

Manually Data Collected (more than 30k tweets) : https://crisislex.org/ : this is the website for manually collecting data collection, It has data with labels and is stored in CSV format. But mostly data is not of our use because disaster held outside India will not gives proper result for creating question set for country disaster
We Collect following data on disaster :

1. Tweets from 4 crises in Italy, labeled by relatedness and type
    a. **Contents**: ~5.6K tweets posted during 4 crisis events (2 earthquakes and 2 floods) in 2009, 2012, 2013 and 2014.
    b. **Sampling method**: tweets sampled by keywords.
    c. **Labels**: ~5.6K tweets (between 400 to 3100 in each collection) were labeled by three annotators according to the type of information they convey (as "damage", "no damage", or "not relevant").
    d. **Data format**: comma-separated values (.csv) files containing the text of the tweets and labels, as well as other tweet fields (user, geo-location, time, etc).
2. Geo-Located tweets from the 2012 Sandy Hurricane.
    a. **Contents**: tweet ids for 6,556,328 tweets, representing all tweets from October 22nd, 2012 —the day Sandy formed— until November 2nd, 2012 — the day that it dissipated.
    b. **Sampling method**: tweets were geotagged and located in Washington DC or one of 13 US states affected by Sandy
    c. **Labels**: no labels. The corpus contains tweets both relevant and irrelevant to Hurricane Sandy (no content based filter was applied).
    d. **Data format**: comma-separated values (.csv) files containing the tweet ID, the time stamp of the tweet, a field indicating whether the tweet contains the word "sandy".

Automated Script for Collection(more than 70k tweets) :

And the rest of the data collected from our script is mostly a disaster that happened in India with no filtering and label, this data was just raw & meta-data which is not suitable for classification and generating question sets.

Mainly these data collected in 2 different format:

1. JSON format

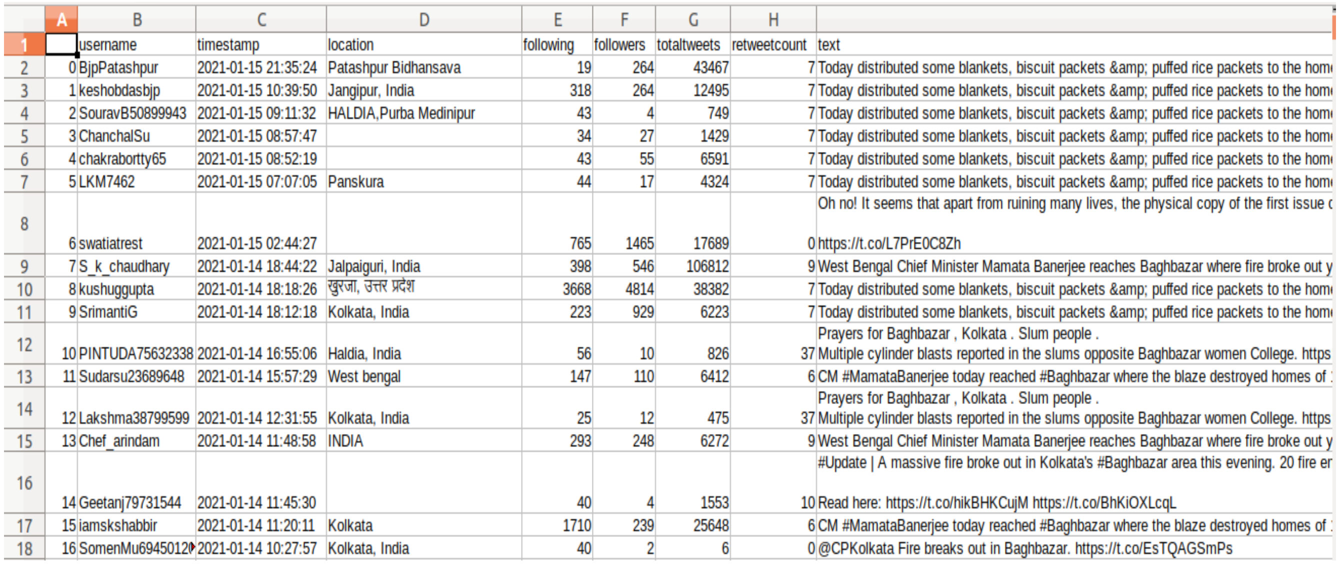
JSON is a generic data format with a minimal number of value types: strings, numbers, booleans, lists, objects, and null. JSON is a good candidate to transmit data across language gaps. JSON data is stored in files that end with the .json extension. In keeping with JSON’s human-readable ethos, these are simply plain text files and can be easily opened and examined.

KEY	VALUE
TIMESTAMP	TWEETS

2. CSV format

A CSV is a comma-separated values file, which allows data to be saved in a tabular format. CSVs look like a garden-variety spreadsheet but with a .csv extension. They help companies export a high volume of data to a more concentrated database, for instance.

USERNAME	TIMESTAMP	LOCATION (if available)	TWEETS



	A	B	C	D	E	F	G	H	
1		username	timestamp	location	following	followers	totaltweets	retweetcount	text
2	0	BjpPatashpur	2021-01-15 21:35:24	Patashpur Bidhansava	19	264	43467	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
3	1	keshobdasbjp	2021-01-15 10:39:50	Jangipur, India	318	264	12495	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
4	2	SouravB50899943	2021-01-15 09:11:32	HALDIA,Purba Medinipur	43	4	749	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
5	3	ChanchalSu	2021-01-15 08:57:47		34	27	1429	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
6	4	chakrabortty65	2021-01-15 08:52:19		43	55	6591	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
7	5	LKM7462	2021-01-15 07:07:05	Panskura	44	17	4324	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
8	6	swatiatrest	2021-01-15 02:44:27		765	1465	17689	0	Oh no! It seems that apart from ruining many lives, the physical copy of the first issue https://t.co/L7PrE0C8Zh
9	7	S_k_chaudhary	2021-01-14 18:44:22	Jalpaiguri, India	398	546	106812	9	West Bengal Chief Minister Mamata Banerjee reaches Baghbazar where fire broke out y
10	8	kushuggupta	2021-01-14 18:18:26	खुरजा, उत्तर प्रदेश	3668	4814	38382	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
11	9	SrimantiG	2021-01-14 18:12:18	Kolkata, India	223	929	6223	7	Today distributed some blankets, biscuit packets &amp; puffed rice packets to the hom
12	10	PINTUDA75632338	2021-01-14 16:55:06	Haldia, India	56	10	826	37	Prayers for Baghbazar , Kolkata . Slum people . Multiple cylinder blasts reported in the slums opposite Baghbazar women College. https
13	11	Sudarsu23689648	2021-01-14 15:57:29	West bengal	147	110	6412	6	CM #MamataBanerjee today reached #Baghbazar where the blaze destroyed homes of
14	12	Lakshma38799599	2021-01-14 12:31:55	Kolkata, India	25	12	475	37	Prayers for Baghbazar , Kolkata . Slum people . Multiple cylinder blasts reported in the slums opposite Baghbazar women College. https
15	13	Chef_arindam	2021-01-14 11:48:58	INDIA	293	248	6272	9	West Bengal Chief Minister Mamata Banerjee reaches Baghbazar where fire broke out y
16	14	Geetanj79731544	2021-01-14 11:45:30		40	4	1553	10	#Update | A massive fire broke out in Kolkata's #Baghbazar area this evening. 20 fire er Read here: https://t.co/hikBHKCujM https://t.co/BhKiOXLcqL
17	15	iamskshabbir	2021-01-14 11:20:11	Kolkata	1710	239	25648	6	CM #MamataBanerjee today reached #Baghbazar where the blaze destroyed homes of
18	16	SomenMu6945012	2021-01-14 10:27:57	Kolkata, India	40	2	6	0	@CPKolkata Fire breaks out in Baghbazar. https://t.co/EsTQAGSmPs

Fig. 3.5 Actual Data Collected

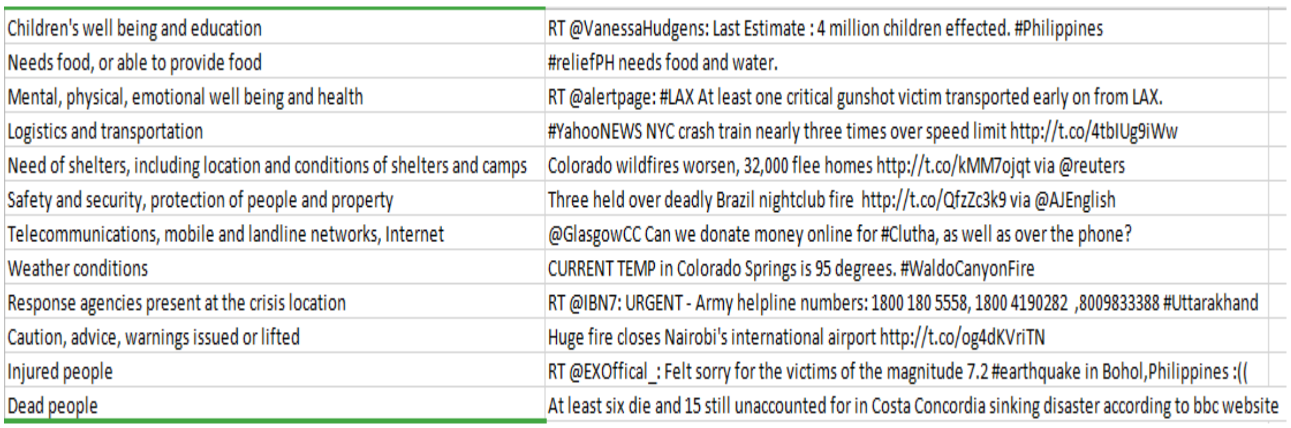
Chapter 4
Methodology

4.1 Data Classification

Data classification is the process of analyzing structured or unstructured data and organizing it into categories based on file type, contents, and other metadata.

Data classification helps organizations answer important questions about their data that inform how they mitigate risk and manage data governance policies. It can tell you where you are storing your most important data or what kinds of sensitive data your users create most often. Comprehensive data classification is necessary (but not enough) to comply with modern data privacy regulations.

The tweets obtained from twitter are further annotated with classes according to their content. Currently, the annotation is done manually. Table below shows the data classification. The dataset being used for testing has been retrieved from ”CrisisLex”, a popular disaster data repository. It has 7312 annotated tweets having 23 different categories.



Children's well being and education	RT @VanessaHudgens: Last Estimate : 4 million children effected. #Philippines
Needs food, or able to provide food	#reliefPH needs food and water.
Mental, physical, emotional well being and health	RT @alertpage: #LAX At least one critical gunshot victim transported early on from LAX.
Logistics and transportation	#YahooNEWS NYC crash train nearly three times over speed limit http://t.co/4tbIUg9iWw
Need of shelters, including location and conditions of shelters and camps	Colorado wildfires worsen, 32,000 flee homes http://t.co/kMM7ojqt via @reuters
Safety and security, protection of people and property	Three held over deadly Brazil nightclub fire http://t.co/QfzZc3k9 via @AJEnglish
Telecommunications, mobile and landline networks, Internet	@GlasgowCC Can we donate money online for #Clutha, as well as over the phone?
Weather conditions	CURRENT TEMP in Colorado Springs is 95 degrees. #WaldoCanyonFire
Response agencies present at the crisis location	RT @IBN7: URGENT - Army helpline numbers: 1800 180 5558, 1800 4190282 ,8009833388 #Uttarakhand
Caution, advice, warnings issued or lifted	Huge fire closes Nairobi's international airport http://t.co/og4dKVriTN
Injured people	RT @EXOffical_: Felt sorry for the victims of the magnitude 7.2 #earthquake in Bohol,Philippines :((
Dead people	At least six die and 15 still unaccounted for in Costa Concordia sinking disaster according to bbc website

Table. 3.5 Classification

We have evaluated the performance of BERT using K-Fold Cross Validation on the annotated data set. The data set has a category corresponding to each tweet. Each category serves as a different class. The model serves as a text-classifier to classify tweets according to their category.

4.2 K-Fold Cross Validation

Cross-validation is a re-sampling procedure used to evaluate machine learning models on a limited data sample.

The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    a. Take the group as a hold out or test data set
    b. Take the remaining groups as a training data set
    c. Fit a model on the training set and evaluate it on the testset
    d. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores.

The results of a k-fold cross-validation run are often summarized with the mean of the model skill scores.

4.2.1 Configuration of K

The k value must be chosen carefully for your data sample.

A poorly chosen value for k may result in a mis-representative idea of the skill of the model, such as a score with a high variance (that may change a lot based on the data used to fit the model), or a high bias, (such as an overestimate of the skill of the model).

Three common tactics for choosing a value for k are as follows:

- Representative: The value for k is chosen such that each train/test group of data samples is large enough to be statistically representative of the broader dataset.
- k=10: The value for k is fixed to 10, a value that has been found through experimentation to generally result in a model skill estimate with low bias and a modest variance.

- k=n: The value for k is fixed to n, where n is the size of the dataset to give each test sample an opportunity to be used in the hold out dataset. This approach is called leave-one-out cross-validation.

4.3 Filtering Data Using Python

Data filtering is the process of choosing a smaller part of your data set and using that subset for viewing or analysis. Filtering is generally (but not always) temporary – the complete data set is kept, but only part of it is used for the calculation.
Filtering may be used to:

- Look at results for a particular period of time.
- Calculate results for particular groups of interest.
- Exclude erroneous or "bad" observations from an analysis.
- Train and validate statistical models.
- Removing unwanted data value or tuples

Filtering requires you to specify a rule or logic to identify the cases you want to include in your analysis. Filtering can also be referred to as "subsetting" data, or a data "drill-down".

Here in this filter code we have written a Python Script named "parser.py", which generally deals with removing retweets and unwanted characters. Also this parser which converts every character in tweets to a subset of ASCII characters. Also removes emojis and additional symbols which are not required in sentences or in training data.

After filtering the data set, we can now go for classification models as this will help us to get more accuracy during classification using the BERT model.

Algorithm for Filtering

The filtering data algorithm is very straightforward. You have to parse the each row in CSV file and exclude the columns which do not contains tweets because for label and classification we only need tweets not username and timestamp and then you have to check every tweet in tweets columns character by character and see if its is ASCII Character then append in new CSV file otherwise skip and remove emojis and unnecessary characters.

```
1. Load CSV File into Python script
2. Run 1 outside loop to read every Column
    a. Run 1 inside loop to read row of every Column
        i. Check if column is of tweets
        ii. Then run a loop to read every character in tweets
            1. If character found is Ascii then append in new CS
        iii. End loop
    b. Append every row in new CSV
3. End loop
```

After filter file will be change from **meta-data.csv → filtered-data.csv**

Chapter 5
Results

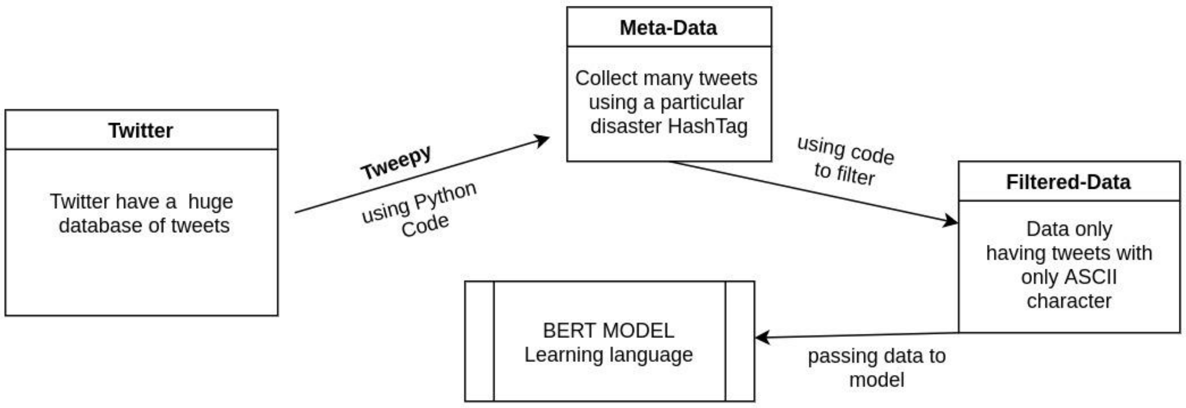
5.1 Training With BERT

The results of the K-fold Cross Validation with the BERT model had an accuracy of 47% which is not satisfactory.

The lesser accuracy obtained may be due to insufficient data sets. The data set consisted of only 7500 annotated tweets. Adding more examples, adds diversity. It decreases the generalization error because the model becomes more general by virtue of being rained on more examples.

Basically this accuracy varies from data to data and the way we label data. Accuracy also depends on the volume of tweets.

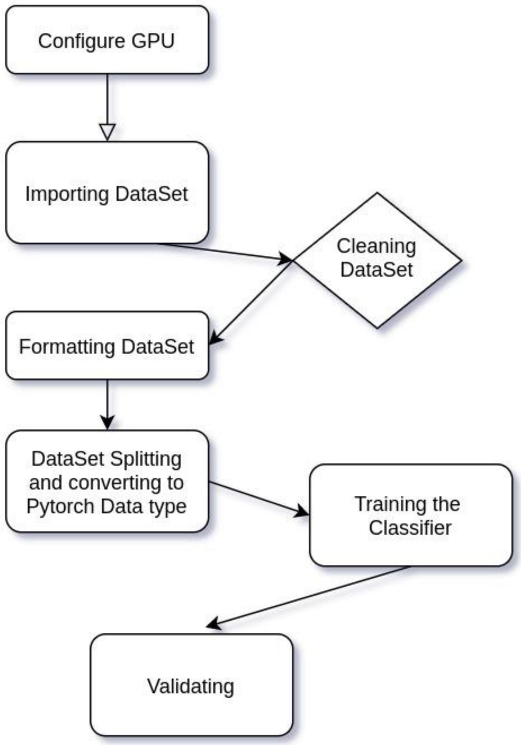
Following is the procedure from collecting data to training the model and validating at the end.


Twitter
Twitter have a huge database of tweets
Tweepy
using Python Code
Meta-Data
Collect many tweets using a particular disaster HashTag
using code to filter
Filtered-Data
Data only having tweets with only ASCII character
passing data to model
BERT MODEL Learning language

BERT MODEL Working on Google Colab

Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

We have written our code to Google Colab and uploaded it to run at the time we need. Because my machine doesn't have better configuration to run Pytorch and Tensorflow.



5.2 Number of labels v/s Accuracy

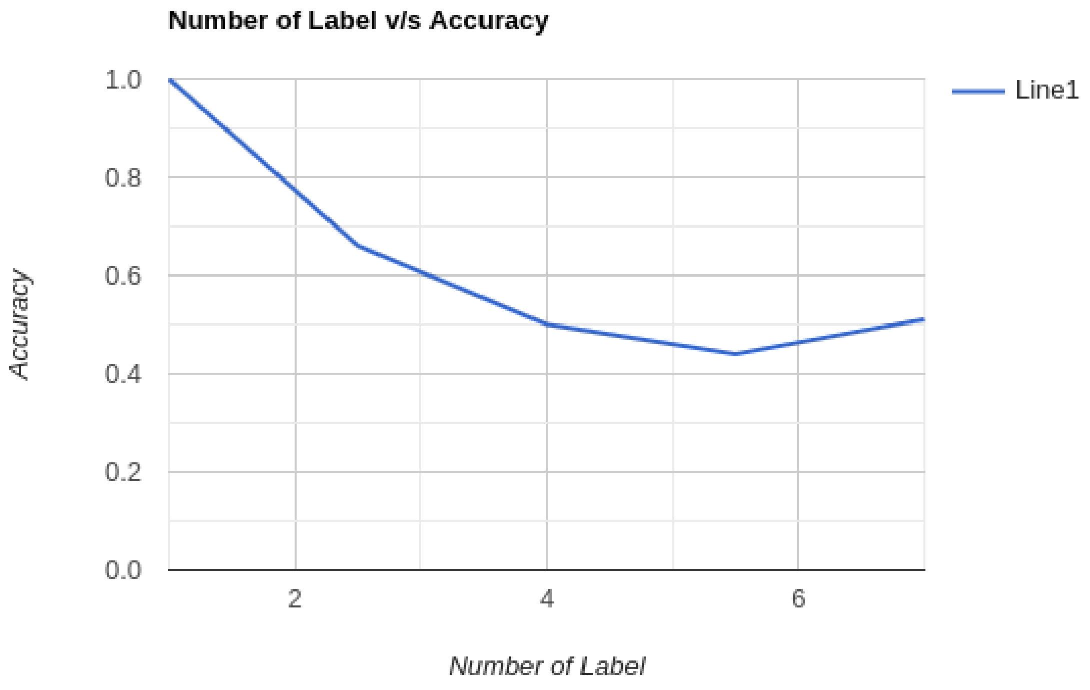
We have taken a dataset having 500 tweets(constant for every number of labels) for this test and training. We analysed the flow of a model having a variable number of labels to classify tweets.

We got the result which is not satisfactory as the graph has a downward slope which means accuracy decreases when the number of labels increases but if we provide or manually plot labels with correct tweets we get approx 50% accuracy from the model.

Here we plot the number of labels varying from 1 to 7 on x-axis and on y-axis we get Accuracy results varying from 0.00 to 1.00 (1.00 means 100% accuracy).

Number of labels	1	2	3	5	7
Accuracy	1.00	0.66	0.50	0.44	0.51

Dataset is having 500 number of tweets as volume which is constant



5.3 Volume v/s Accuracy

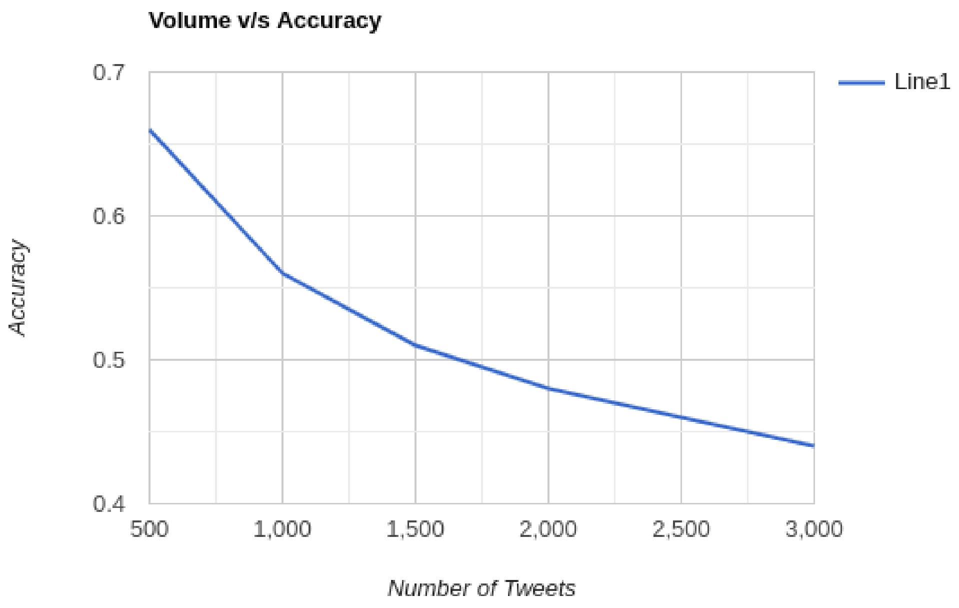
In this training and result analysis we have taken a dataset having a different volume of data (volume is measured as the total number of tweets the dataset is having), and we have taken a constant number of labels in every dataset i.e 7 unique labels.

We have analysed that our line graph is having downward slope i.e. accuracy decreases when number of tweets (volume) increases, but it remains constant after 3000+ tweets which is 44%.

On the x-axis we have plot volume of data varying from 500 tweets to 3000+ tweets, on the other hand on y-axis we got accuracy of the model as result varying from 0.00 to 1.00 (1.00 means 100%)

Volume Of Data (number of tweets)	500	1000	1500	2000	2500	3000+
Accuracy	0.66	0.56	0.51	0.48	0.46	0.44

DataSet having 7 unique label



After the result and analysis we have come to the conclusion that if we have a reasonable amount of labels attached with correct tweets and proper volume of data we will get good accuracy as a result by the BERT model.

All the test and training and validation is completely down on Google Colab by running python script

Future Work

6.1 Collecting More Data

In our further work, we will collect and annotate more data sets from various sources to train our model better. As BERT model requires
Huge amount of data to work with high accuracy. Also we can develop better Neural Language Model by giving more than 2-3 passes with training data to make the Model learn language better.

6.2 Fine Tuning BERT

In the fine-tuning training, most hyper-parameters stay the same as in BERT training. The BERT team has used this technique to achieve state-of-the-art results on a wide variety of challenging natural language tasks

There are Two phases in BERT. First is to learn the language so we have to provide training data to make BERT model learn the language and work for some specified task

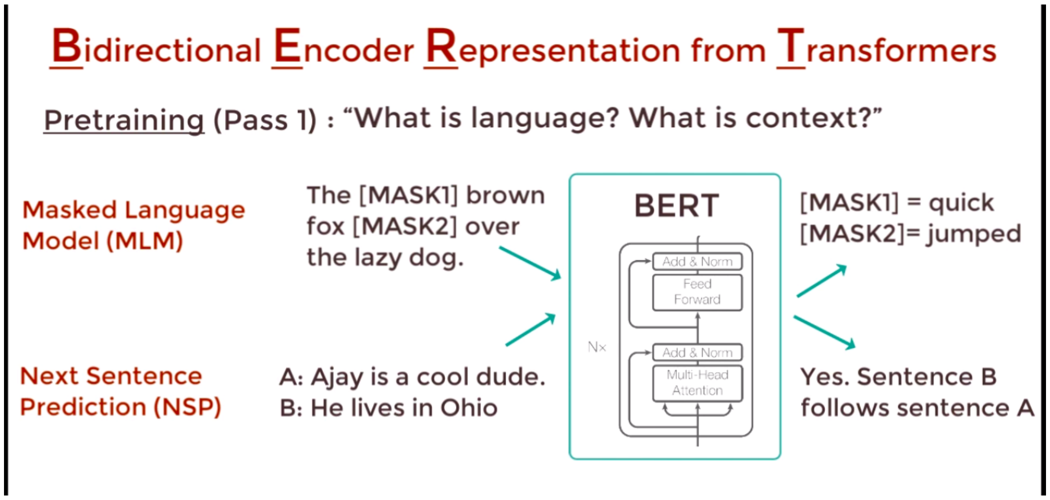
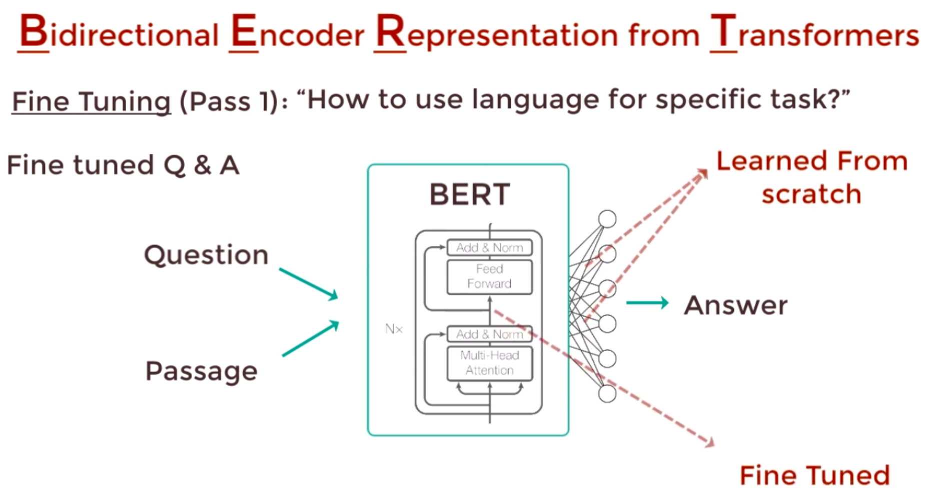

Bidirectional Encoder Representation from Transformers
Pretraining (Pass 1) : "What is language? What is context?"
Masked Language Model (MLM)
The [MASK1] brown fox [MASK2] over the lazy dog.
BERT
Add & Norm
Feed Forward
Nx
Multi-Head Attention
[MASK1] = quick
[MASK2]= jumped
Next Sentence Prediction (NSP)
A: Ajay is a cool dude.
B: He lives in Ohio
Yes. Sentence B follows sentence A

FIG. 6.2 Pretraining

Second is to do some tuning to make the model work for some specified task which may be providing questions & paragraphs(data set) as input and getting answers as output.



References

1. https://github.com/google-research/bert (BERT Repository)
2. https://ieeexplore.ieee.org/abstract/document/7963782/ (for lstm network)
3. https://ieeexplore.ieee.org/abstract/document/956751/ (for transformer network)
4. https://crisislex.org (For Disaster Dataset)
5. https://simpletransformers.ai/docs/
6. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 (for BERT explanation)
7. https://gist.github.com/sxshateri/540aead254bfa7810ee8bbb2d298363e (for tweepy)
8. https://pypi.org/project/twint/ (for twint details)
9. https://github.com/surajgupta09/TwitterBot (for code and data)
10. Links for DATA SETS
    a. https://drive.google.com/drive/folders/115lck9uMowrVrJpUS7moYvfuGoDSEkMW?usp=sharing
    b. https://drive.google.com/drive/folders/1Mw_Gykv2FN6C98CA6fusm698ob0cDErr?usp=sharing
    c. https://drive.google.com/drive/folders/1JCti9rWne9boT7PiMceLOioCJGQuAt_i?usp=sharing