Social Media Analytics for Healthcare Surveillance using Text Mining

Submitted May 2020, in partial fulfillment of
the conditions for the award of the degree **BSc Computer Science.**

Yuyang Liu
16522049

Supervised by Heng Yu

School of Computer Science
University of Nottingham

Contents

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and related works

Disease control and prevention is vital for the whole society. Traditional surveillance method adopted by the Centers for Disease Control and Prevention (CDC) is, scrutinizing outpatient records from hospitals and virological test results from laboratories, which notices the disease after it actually occurred [55]. However, if there is no forecast and hospitals are ill-prepared for a rush of patients, the reception of in-time treatment will be affected [21], other severe consequences can also be imagined. Therefore, a robust disease forecast system is needed. To predict the outbreak of disease in advance, massive efforts have been put. [15] monitored the changes of Realistic Contact Networks (RCNs) to predict the dynamic movement of disease, [14] used machine learning with previous illness records to predict the future outbreak, while some researchers tried to build a prediction system based on data from social media. According to [37], social media contains information related to healthcare, individual health issue, symptom. [24] shows that spikes in flu queries and disease breakout coincide. However, since queries has little or no limitation and even don not need an account, they cannot be regarded as reliable data [55]. Other social media platforms such as Twitter and Facebook have proven their value for Big Data analyze. Twitter data has been found to be useful for public health applications [19], including: (1) monitoring diseases, (2) public reaction, (3) outbreaks or emergencies, (4) prediction, (5) lifestyle, and (6) geolocation of disease surveillance [4]. In addition, social media is prompt. According to [21], over 645 million active Twitter users collectively post an average of 58 million tweets (micro-blogs no more than 140 characters long) per day in 2017, and the number is still growing. A practical example is that researcher use Twitter predicted flu outbreaks 1–2 weeks ahead of CDC's surveillance average [56]. [21] also showed Twitter data aligns with CDC's outpatient records. All the information I have read so far proves that such data is valuable in healthcare surveillance.

1.2 Motivation

Previous work relying on social media has successfully proposed some novel methods. [53] and [54] utilize time-series analysis on single geography, [21] provided a generalized solution to identify how contagious diseases diffuse across geographies. However, time-series analyze can be inaccurate in this scenario (they rely merely on the statistics of social media instead of its content). For example, top search about a certain disease could result from a celebrity's illness [55]. Similar experimental result showed in [21]. During festivals, the number of tweets decreased, and the prediction accuracy was affected. Obviously, such results are not robust enough to be applied in the real world. It therefore makes sense to find a new method to enhance the whole system. In this project, I will start from previous works, and use different techniques such as machine learning, heuristic function, NLP to extract textual implication containing in such data and create a more feasible solution.

Chapter 2

Aims and Objectives

The general aim of this project is analyzing social media data to surveille healthcare condition. It can be detailed as follow:

1. Healthcare-related textual information should be extracted and modeled for the purpose of healthcare surveillance.

2. Robust predictive models are required for accurate forecasting of disease outbreaks and hospital emergency visits based on ML techniques.

The key objectives of this project are:

1. Collecting data of a certain social media platform. This data can be either extracted from an existing data set or crawled from that platform. If existing data set can't meet my requirement, my own data set will be made.

2. Filtering data. In this process, a filtering rule should be established, possible solutions include using related verbal list, using NLP to classify raw data

3. Designing algorithm that can successfully predict the outbreak of diseased and its propagation direction with high accuracy, and implementation. This includes (1) setting a benchmark to evaluate the accuracy; (2) finding a suitable model maxing the accuracy.

4. Experiment. Testing and evaluating this model, and compare it with others' work. In this stage, re-implementation may be needed.

5. Result visualization. Since the outbreak is dynamic, visualizing the result can be more readable.

Chapter 3

Design

3.1 Basic assumptions

The whole project is designed based on two basic assumptions: (1) social media data can be used to predict the outbreak of diseases; (2) available data contain sufficient information to show the relationship between input and output.

3.2 Overall design

Generally, our system can be divided into two components: (1) signal extracting: extract healthcare-related topics from social media; (2) modeling: build a predictive model based on extracted signals. Treat our system as a Blackbox, the input is the metadata extracted from social media platforms while the output is the prediction of diseases (Figure 3.1).

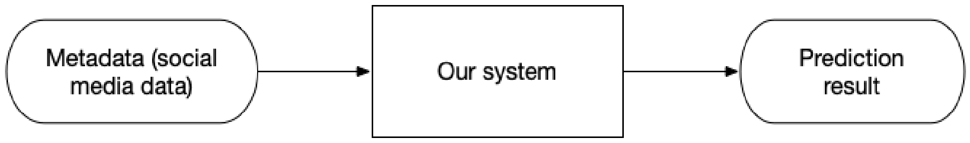


Figure 3.1: Blackbox

The aims of overall design involve building a complete process where the performance is reproducible, the subprocesses can be adjusted and the outcome can be easily understood and visualized. Based on it, we separate the Blackbox into 6 independent components: Data Collector, Preprocessor, Classification Model, Clustering Model, Evaluation layer, visualization. The overall structure can be seen in Figure 3.2, the square represents component while ellipse represents data.

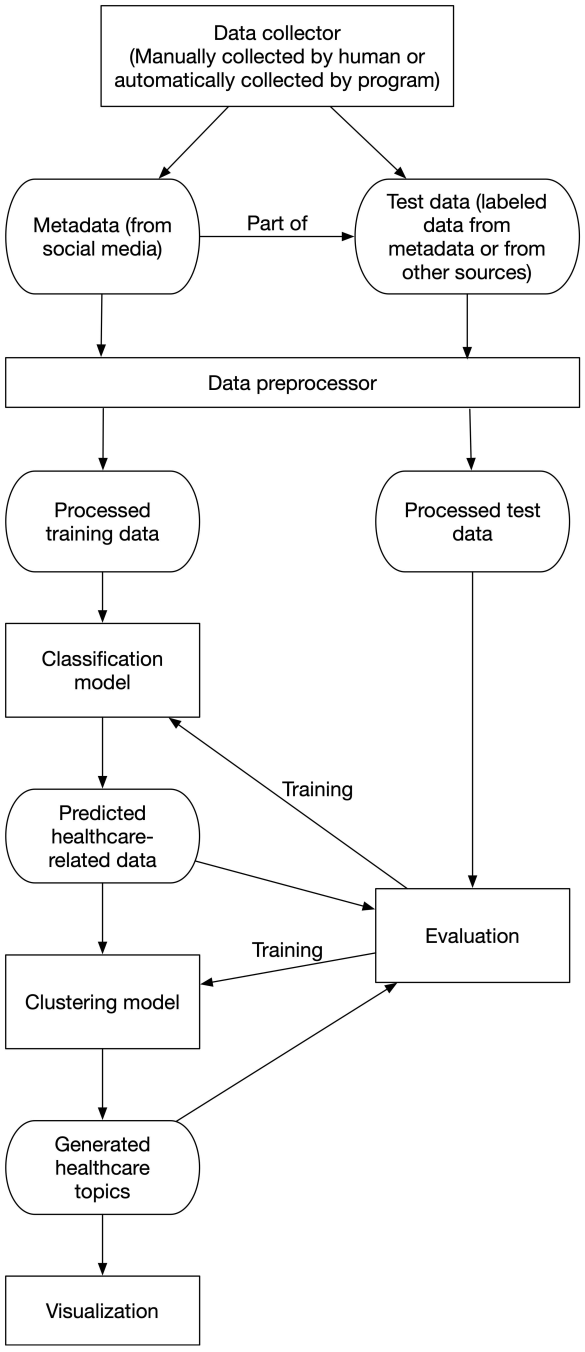

Data collector
(Manually collected by human or automatically collected by program)
Metadata (from social media)
Part of
Test data (labeled data from metadata or from other sources)
Data preprocessor
Processed training data
Processed test data
Classification model
Training
Predicted healthcare-related data
Evaluation
Training
Clustering model
Generated healthcare topics
Visualization

Figure 3.2: Overall design

3.3 Basic components

This section show the preliminary design of our components, including the basic functional requirements of each part and the possible methods we may adopt. Following subsections represent stages(pipelines) of our system respectively, the general procedure design is inspired by [22].

3.3.1 Data collector design

Social media data is the input of the whole system, but according to the different social media platforms and various sources of data (such as extracted from official API or download from open-source dataset), the structure of data and methods of collecting data can be diverse. Therefore, we design an interface which can collect and integrate different metadata. In this project, we focus more on modeling and algorithm design rather than a complete system, therefore, we choose one certain social media platform. In addition, to evaluate the accuracy and performance of our model, we need a ground truth dataset (if available). Here the interface should at least collect these two dataset from at least one source respectively.

Functional requirements of this component are:

1. Collect and store social media data from at least one source
2. Collect and store ground truth data from at least one source, or manually label some data from metadata

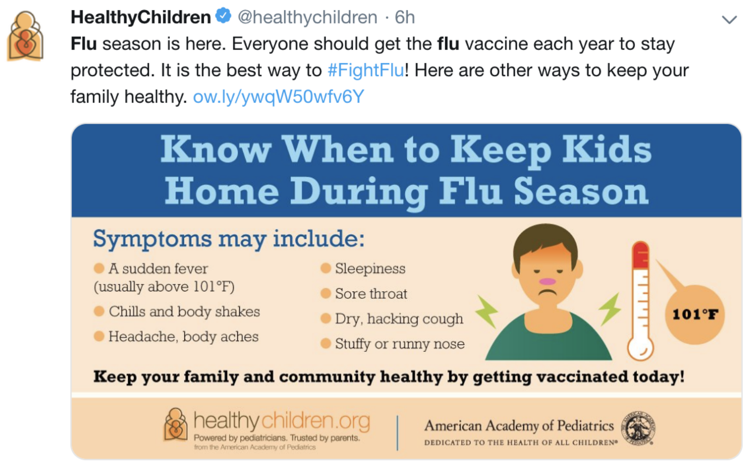
3.3.2 Data preprocessor design

Data collected by stage 1 are metadata, which could be unstructured and irrelevant to this project. We subdivide this step into smaller steps:

1. Unify data structure: if one dataset comes from different sources (for example, facebook data extracted from API and from web spider). To pass these data into functions of

later steps, a unified format is required. In addition, dataset could contain information that won't be used by our algorithm, ignore such information when unify data can save storage space. In our design, both unified structures of social media dataset and ground truth dataset should be implemented (see section 4.3).

2. Text regularization: social media dataset could adopt different coded format (such as ASCII and unicode), here we decide to use utf-8 encoding, which is wildly used in the Internet. The collected data could contain special symbols, unknown characters, URL links and emoji, pictures, videos (See Figure 3.3). In this project, we focus on pure text, thus information such as URL links, pictures and videos will be ignored. Inspired by [72], emoji and some pecial symbols can be transformed in to text based on standard transformation tables. We will adopt such transformation to keep maximum valuable data.


HealthyChildren @healthychildren · 6h
Flu season is here. Everyone should get the **flu** vaccine each year to stay protected. It is the best way to #FightFlu! Here are other ways to keep your family healthy. ow.ly/ywqW50wfv6Y

Know When to Keep Kids Home During Flu Season
Symptoms may include:
A sudden fever (usually above 101°F)
Chills and body shakes
Headache, body aches
Sleepiness
Sore throat
Dry, hacking cough
Stuffy or runny nose
101°F
Keep your family and community healthy by getting vaccinated today!
healthychildren.org
Powered by pediatricians. Trusted by parents.
American Academy of Pediatrics
DEDICATED TO THE HEALTH OF ALL CHILDREN

Figure 3.3: Tweet with URL and picture (screenshot from Twitter)

3. Data filtering (classification): after the structuralization, the dataset can be used for analysis. However, not all the data contain information we want. This step will filter out irrelevant data of both social media dataset and ground truth dataset and reserve data that be considered useful. We will set inclusion rules and classifier to filter the data and label them (see section 4.4). The method we adopt to train a classifier is similar to

how we build our prediction model. The detailed methods of text tokenization,encoding and building neural network model can be found in stage 3 of this section.

4. Location extraction (possibly): in this project, we need data containing time of creation, geographic information to create our diffusion model. Data without such information can be used to train a classifier in the next step. The time of creation is contained in most sources (all the datasets we search so far provide temporal information). However, based on user's setting, some data don't contain geographic information (users can hide their private information if they wish). In addition, some platforms allow users name their own location (such as Wechat, users can assign personalized name to their location), or use a fake one. Platforms can adopt different standards of placename. Apart from that, even a user provide authentic private location, it still can't be guaranteed that he was in that place when posted tweets. All of such conditions bring noises in the dataset. Use the method we deal with unstructured metadata as reference, here we plan to the same solution: regularize the geographic information, set a standard in our project. In term of customized placename, we will design a function trying to map it to our standard (see Chapter 4). However, it worth noting that, the percent of social media with geographic information is comparably samll, which may not be sufficient to build a reliable predictive model (and this design could be changed based on it).

Functional requirements of this component are:

1. Must unify a data structure of social media dataset, and can integrate datasets (if more than one sources) into the same structure

2. Must unify a data structure of ground truth datasets, and can integrate datasets (if more than one sources) into the same structure

3. Must regularize all the text in the integrated dataset

4. Must extract geographic information of each data if available

5. Must regularize all the extracted geographic information into a unified format

3.3.3 Classification Model design

As can be seen in figure 3.2, the final model is designed as an ensemble of two models: a classification model and a clustering model. Classification model is help to screen out data that is irrelevant to healthcare, and hence we will use labeled data to fed it. Therefore, the model follows the the common process of solving NLP tasks.

1. Text tokenization: neural network can only receive tensors as input, therefore, the first step of this stage is transforming the textual data into tensor, which is called tokenization (a token is a single unit extracted from text) [17]. There are three basic word separation strategies: (1)split text into single words, transform each word into a vector; (2)split text into single character, transform each character into a vector; (3) extract n-gram of words or characters, transform each n-gram into a vector (a n-gram is a set of sequential words or characters), the resulting set is called bag-of-words [17]. Bag-of-words can't record the order of words in the original text, therefore, this method is wildly used in shallow-layer model instead of deep learning model. Extracting n-gram is a feature engineering, which is inflexible and unstable. Here we adopt the first strategy, the feature extraction procedure will done by our deep learning model.

2. Text encoding: the procedure that transform token into vector is called encoding. There are two most common methods. One is one-hot encoding, which assign each token a unique integer i, transform i into a binary vector (only contains 1 and 0) of length N (N is the size of token list), only the ith element is 1, others are 0. This method returns a high dimensional sparse vector (20000 dimensions or more), since each token takes one dimension. Anthoer encoding method is word embedding, which is learnt from dataset, returns a low dimensional intensive vector. The idea behind this method is that: the geometrical distance of two token should base on their relation (synonyms should have

shorter distance than antonyms), and the vector's direction should have sense. For example, the vector of word "king" plus the vector of word "female" should return the vector of word "queen" [17]. Therefore, we can't assign each token a vector randomly. In addition, for different tasks, the embedding space could be diverse, the embedding space used for sentiment analysis may not fit argot detection. In our design, we will try both of these two methods, and for the second, we will train a embedding space based on our dataset.

3.3.4 Clustering model design

Some topics could be relatively new or have few samples, it's hard to label the classes of them in advance. Therefore, the classification model could not predict all minor topics related to healthcare. For this reason, we decide to use one more unsupervised model that helps to detect health events. Classification is in charge of deciding whether a given document contains information about healthcare, while the clustering model groups the filtered data in a time slice (such as in one day) and find the hidden topics. The functional requirements of this model are:

1. Must take the filtered data from the classification model, and group them into clusters
2. Must assign a interpretable topic to each cluster

3.3.5 Evaluation layer design

In our overall design (Figure 3.2), there are two cycles linking this layer and two models respectively. This doesn't mean that those two model share the same evaluation criteria. In deed, this layer is abstract, it is designed to show that we will evaluate this two models respectively.

For the supervised classification model, according to [17], there are 8 common methods can be adopted to evaluate the model: (1) confusion matrix; (2) accuracy; (3) precision; (4) recall; (5) F1 score; (6) ROC curve; (7) AUC (Area Under Curve); (8) PR curve. Especially, accuracy and recall are wildly used in class-imbalance problem (our task is class-imbalanced). Following are formulas of (2)(3)(4)(5), where TP, TN, FP, FN represents true positive (the number of cases correctly identified as required), ture negative the number of cases correctly identified as not required, false positive (the number of cases incorrectly identified as required), false negative (the number of cases incorrectly identified as not required), respectively:

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN}$$

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

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

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

Once the socre (target) is defined, we must adopt a method to assess the result. There are 3 common methods: (1) Hold-out method (test set estimation); (2) K-Fold Cross Validation; (3) Repeated k-fold Validation. The first method works by randomly divided dataset into two mutually exclusive subsets, the training set (often $\frac{2}{3} to \frac{4}{5}$ of the original set) and testing set. It is simple to implement but will be severely affected by the size of subsets. If the training instances is far more than testing instances, the evaluation result is unreliable, but in reverse, the model will lose fidelity. In addition, this method is unsuited to small sample sizes, since it can't make full use of data [17, 43]. Second method partition data into k separate subsets of similar size. Each subset will be used as testing data in turns (k times) while left subsets will be used as training data, the final socre is the mean of all rounds. It can be regarded as a kind of hold-out method with the ability to exploit more data and provide higher reliability [43]. The third one is used when the available data are too fewer while high prediction accuracy is required. It repeat the second method and calculate average score [17].

For the clustering model, there are several criteria, such as Silhouette Coefficient. However, we care more about the interpretability of generated topics rather than the grouped clusters. Here we decide to use coherence of topics to evaluate our model. The details can be found in section 5.3. Functional requirements of this component are:

1. It should provide different evaluation indexes (Accuracy, Recall, etc.)to assess the prediction outcome.
2. It should provide different evaluation methods (K-Fold, Hold-our, etc.) to assess the prediction outcome.
3. Must choose a best combination of methods to evaluate prediction result based on ground truth dataset.
4. Must set a baseline (or target) to stop training.
5. The evaluation result must correlate with human judgement.

3.3.6 Visualization design

The final result may not clear and meanful for users. Visualization can help users/researchers to figure out the potential information of data/result, such as its feature, pattern, trend and relationship [26]. There are various visualization techniques for different sceniros, purposes and data/input, such as 2D display (bar chart) and 3D display (cloud vapor), in addition, if the prediction is real-time, the visualization could be dynamic. In this project, visualization is used in the last stage, therefore, we can assume the the input is stable and predictable. In addition, the prediction is numerical, according to [26], geometric representing methods could be used, such as scatetr plot, lines etc. The final method will be adopted based on the experimental result.

Functional requirements of this component includes:

1. It should integrate the whole process and choose an approach to display the result.

Chapter 4

Data collection and processing

In our system design, two types of dataset are needed. The first is the social media textual data, which is the most vital to this system, called social media dataset. It contains the implication of potential outbreak of diseases or other healthcare-related information, all the prediction will be made based on it. In addition, to evaluate the model convincingly, actual records of diseases and additional labeled data are needed to serve as benchmark.

4.1 Social media dataset

Researchers have used the Web as sources of syndromic surveillance for years, including Google Flu Trends (display the statistics of daily query logs related to influenza), Twitter, Facebook [44]. In this project, we choose one certain social media platform to test our algorithm, that is Twitter. More specifically, we focus on tweets expressed in English. These choices based on the characteristics of Twitter:

- Providing location: In our design, the posting location of each post is required. And we found that Twitter provides such information. According to research conducted by [25] in 2016, about 1.6 percent of Twitter users opted in Twitter's location-sharing service.

- Availability: Most tweets are available for research. According to [72], around 95% of Twitter users opted in sharing their tweets with public, meaning their tweets can be searched and filtered by keywords without their permission.

- Comparability: In our research, we found that most related works used data from Twitter, which means that Twitter dataset can act as a benchmark to evaluate our algorithm.

- Timely: According to [72], each tweet is received within seconds of their creation.
- High user volume: According to [25], about 21% of American citizens use Twitter.

4.2 Benchmark

For this project, the evaluation are conducted from multiple aspects: (1) whether a new document can be correctly identified as health-related or not; (2) whether the filtered documents can be properly grouped and get interpretable topics (including find new health event); (3) whether the trend shown in social media follows the actual diagnostic recording. First one aims to evaluate the performance our model, therefore, reliable healthcare-related tweets are required. The benchmark dataset is from the official Twitter account (such as CDCHealth, BBCHealth) and health-related Hashtags. To evaluate the criterion, labeled data are required, but they are not necessary to be health-related. The third one can help to verify the trustability and informativity of social media, real disease data are required. Centers for Disease Control and Prevention (CDC) [12] is a well-known nation's health protection agency recording diseases (such as flu, heart diseases, COVID-2019) and conditions over the US. Data published by CDC are comparably reliable. In this project, we focus on one disease, flu. The benchmark used in this project is Fluview [13], a weekly-update, influenza surveillance report of the U.S. published by CDC. Such report is a collaborative effort between CDC and its many partners in state, local, and territorial health departments, public health and clinical laboratories, vital statistics offices, healthcare providers, clinics, and emergency departments [12]. CDC maintains a surveillance network called Influenza-like Illness Surveillance Network (ILINet), which collect information on outpatient visits to health care providers for influenza-like illness [12]. We choose CDC's Fluview as our benchmark because of its:

- Reliability: ILINet collects data from about 2600 outpatient healthcare providers across the U.S. weekly [12].

- Accessibility: All the reports of Influenza-like Illness (ILI) can be accessed by public.
- Comparability: The dataset maps the activity of ILI into levels between 1 to 10, which can be used as labels or targets of our training data.

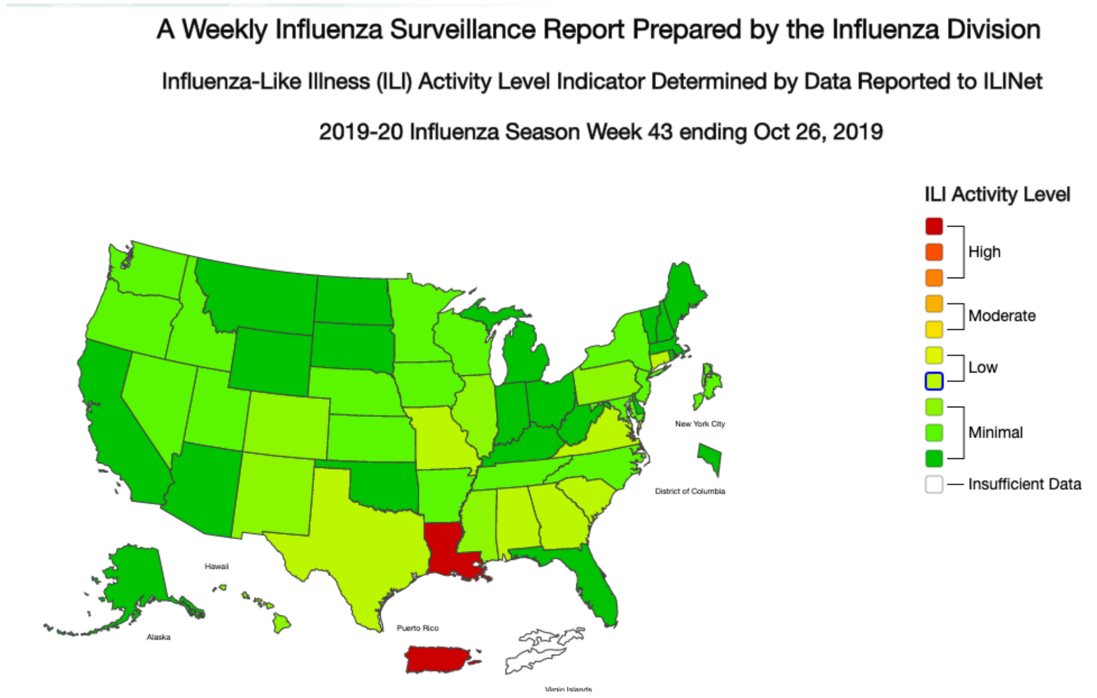

A Weekly Influenza Surveillance Report Prepared by the Influenza Division
Influenza-Like Illness (ILI) Activity Level Indicator Determined by Data Reported to ILINet
2019-20 Influenza Season Week 43 ending Oct 26, 2019
ILI Activity Level
High
Moderate
Low
Minimal
Insufficient Data
New York City
District of Columbia
Hawaii
Alaska
Puerto Rico
Virgin Islands

Figure 4.1: Influenza Season Week 43 ending Oct 26, 2019, Source: Fluview

Figure4.1 shows the ILI activity levels across the U.S. in Week 43, 2019. It can be seen that the there are 10 levels divided into 3 categories, each level is assigned a unique color.

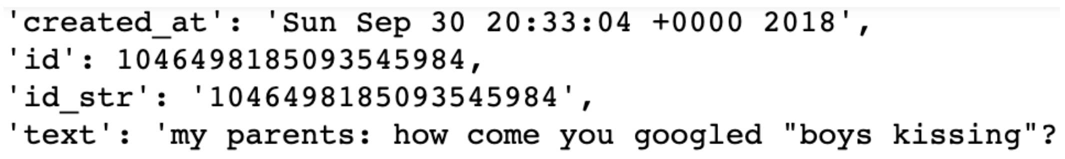
4.3 Data collection

This section shows the detailed methods of how we collect data and unify the data structure.

4.3.1 Twitter dataset collection

In this project, we collect Twitter data from Internet Archive [28], a non-profit digital library of millions of free books, movies, software, music, websites. It contains daily tweets from Feb

2011 to Jul 2019 (Accessed: Dec 08,2019). All the data are Spritzer version (roughly 1% of the whole tweets) grabbed from the general twitter stream The number of tweets collected in Oct 05 2018 is 4273031, in Oct 04 2018 is 4337327, in Oct 01 2018 is 4317376, on average is above 4 million per day. All the tweets are stored in json files. Such data volume is sufficient for our research and its data structure is convenient to use. More important, it contains the information we need, which mentioned in section 4.1. Figure 4.2 shows part of the information those json files contain (geographic and linguistic information are contained but not listed here). In our project, we mainly focus on tweets posted during 2018.



```
'created_at': 'Sun Sep 30 20:33:04 +0000 2018',
'id': 1046498185093545984,
'id_str': '1046498185093545984',
'text': 'my parents: how come you googled "boys kissing"?
```

Figure 4.2: Screenshot of Archive's Twitter data

4.3.2 Benchmark dataset collection

trackmyhashtag (Accessed: March 25, 2020) provides more than 60K free tweets with hashtags "COVID-19" from 1st Dec, 2019 to 28 Jan 2020, and historical tweets posted by 16 official health-news Twitter accounts from 2012 to 2015. Those data can be treated as real healthcare-related tweets and they are used to evaluate the supervised model in our design. [49] published 6 labeled short text datasets for unsupervised clustering algorithm on their GitHub (Accessed: March 25, 2020). We use part of them to evaluate our unsupervised model. Fluview data can be downloaded from official websites of CDC [13] (Accessed: Dec 08,2019), user can customize the data they want to download (the time span of reports).

4.4 Data Preprocessing

Data preprocessing is the first stage of a typical text classification framework. According to experiment conducted by [66], different combination of preprocessing methods can influence the accuracy of prediction. However, there is no best combination for all tasks. Some strategies can improve classification success of certain tasks while lower that of others. Figure

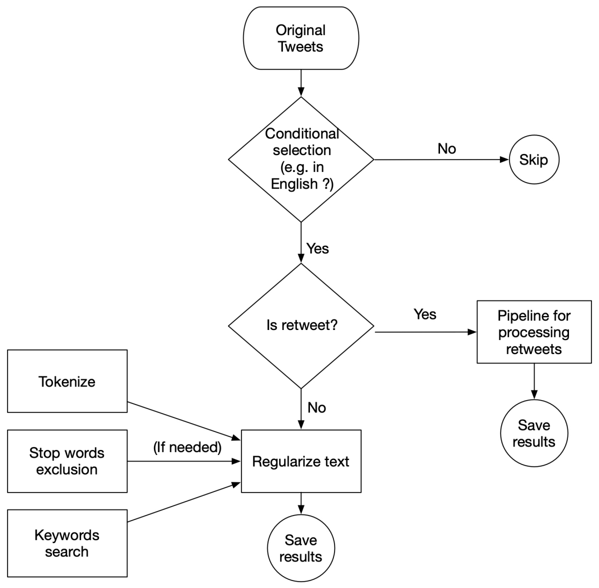


Figure 4.3: Common pipeline of text preprocessing

4.3 shows some common steps used in NLP for preprocessing textual data. Following subsections explain the details of our preprocessing methods designed for this project.

4.4.1 Unify data structure

In section 3.3.1, we mentioned a unified data structure of our datasets. The aim of this stage is to unify and regularize all the metadata before analysis.

The Fluview dataset is well structured (see Figure **??**), and can be used in our system directly (we used its “STATENAME”, “ACTIVITY LEVEL”, “ACTIVITY LEVEL LABEL” and “WEEK” columns). One point should be noticed here is that the STATENAME used in Fluview dataset can be different from that in Twitter (Twitter' users can set their own location), therefore, we created a unified name list of states and a function designed to regularize different geographic information.

Our social media dataset's structure is built on Twitter's official data structure [63], and we only retain the information we may use. It is a hashmap with 6 keys: "created_at", "text", "location" and "coordinates", "place", "hashtags". We regularize the time into format Year/Month/Day (2018/10/01) and store it in "created_at" keys, exclude other information. Note that in the metadata, there are massive "deleted" tweets, which contain no textual information, and we removed all such data. "hashtags" is not a original key in metadata. Hashtags are wildly used in social media and it explicitly indicate topics of tweets. With them, modeling and evaluation could be more accurate. According twitter's official document [63], there are two classes of geographical metadata in Tweet data: (1) Tweet location, which is available when users share location at time of Tweet; (2) Account Location: a free-form character field set in user's profile and may or may not contain metadata that can be geo-referenced. "location" attribute stores the user-defined placename. "coordinates" attribute provides the exact location of a tweet (in long-lat order) but has no placename and only available when the location is assigned. "place" attribute is always present when a Tweet is geo-tagged, and it contains Twitter "place" with a display name and type. Here we keep all these three keys, and "place" key gets the first priority when we extract location of tweet. Figure 4.5 shows a sample of Twitter "place". Note that most data don't contain "location", "coordinates" and "place" keys, which is expected in section 4.1.

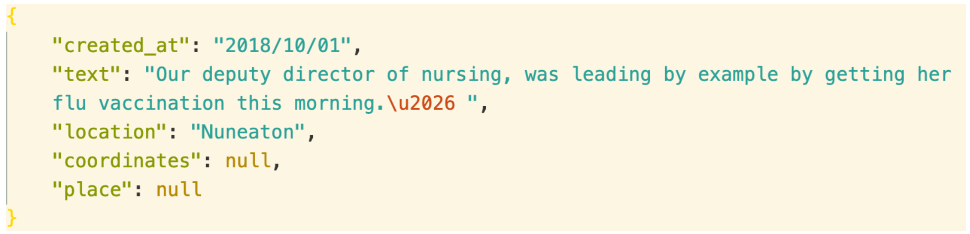
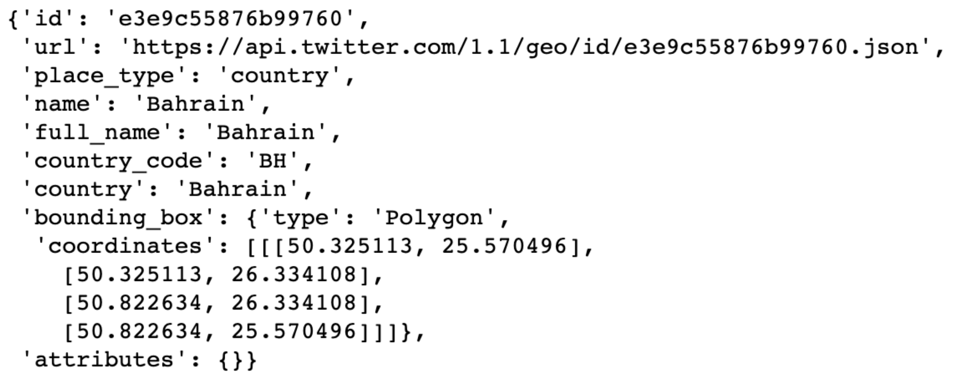

```
{
    "created_at": "2018/10/01",
    "text": "Our deputy director of nursing, was leading by example by getting her
    flu vaccination this morning.\u2026 ",
    "location": "Nuneaton",
    "coordinates": null,
    "place": null
}
```

Figure 4.4: Screenshot of unified structure of social media dataset



```
{'id': 'e3e9c55876b99760',
 'url': 'https://api.twitter.com/1.1/geo/id/e3e9c55876b99760.json',
 'place_type': 'country',
 'name': 'Bahrain',
 'full_name': 'Bahrain',
 'country_code': 'BH',
 'country': 'Bahrain',
 'bounding_box': {'type': 'Polygon',
  'coordinates': [[[50.325113, 25.570496],
    [50.325113, 26.334108],
    [50.822634, 26.334108],
    [50.822634, 25.570496]]]},
 'attributes': {}}
```

Figure 4.5: Screenshot of Twitter's place object from our dataset

4.4.2 Text regularization

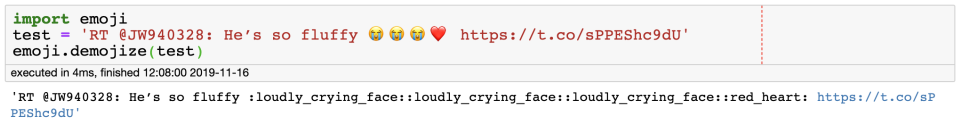
As mentioned in our design (3.3.2), the original text is unstructured, which can not be put in to our model directly and hard for labeling. In addition, each dataset has its own structure, meaning that there is no common regularization rule can be applied to all tasks. Here our regularization rule is built on the observation of our dataset and on our judgement. Followings are common formats we observed in our dataset, expressed in regular expression (Python version):

- retweet: retweeting is a way of forwarding content to others (like forwarding an email). It starts with a "RT " pattern.

- @: in social media, @ refers to a person/group in a conversation and has no meaning for analysis. Its regular expression is "+[\S]*".

- URL links: some tweets contain URL links starting with "http", "ftp" or "https", which can't contribute to our classification. Its regular expression is:
  "http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+"

- emoji: emojis in our dataset are encoded in unicode, inspired by [72], some of them can be translated into their name based on emoticon dictionaries. Through our search, we find a Python library called emoji(version 0.5.4), which embedded the full emoji list from [65] and can help us to translate emoji into text through function call [60]. Figue 4.6 shows a example how to use this library, the translated emojis are embraced by ‘:’ signs by default. In our implementation, each translated emoji is assigned a prefix ‘emo_’ to identify it, and we separate emojis with a blank space for split convenience.



```
import emoji
test = 'RT @JW940328: He’s so fluffy 😭😭😭❤️ https://t.co/sPPEShc9dU'
emoji.demojize(test)
```

```
executed in 4ms, finished 12:08:00 2019-11-16
```

```
'RT @JW940328: He’s so fluffy :loudly_crying_face::loudly_crying_face::loudly_crying_face::red_heart: https://t.co/sP
PEShc9dU'
```

Figure 4.6: Example of using emoji library

- e-mail address: we exclude e-mail address in our data, whose regex(can match most e-mail addresses) is: “[a-zA-Z0-9_.+-]+[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$”

- html entities: a html entity can be regarded as escape characters used in html (eg. &quot; represents “” sign). We use the Python’s standard library “html” to translate it.

- non-Latin characters: characters that are not in Latin are meaningless, we exclude them with regex “[Â-Za-z0-9]”

- hashtags #: In social media, hashtags explicitly indicate topics of documents. Its regex can be written as “[#]1(
w+)”

For convenience, all characters of processed documents are in lower case.

4.4.3 Manually screening

We have more than 4 million tweets per day in our dataset, it’s obviously not all of them are relevant to our task. Based on our initial design, we accept tweets written in English. While reviewing on the dataset, we found that there are massive retweets in it (18682273 of 134704400, roughly 13.87%). Defined by Twitter [63], retweeting is forwarding content wrote by other users (like forwarding an email). Although retweets can be used in some tasks such as sentiment analysis (mainly focus the counts of being retweeted) [48], we don't think it will contribute to our prediction. As stated by [32], retweets don't have user's own view and should be removed in data cleaning. We need tweets that can show the health condition of the user or of people the user cares. In addition, retweets don't have any geographical information [63], which is the key component in our diffusion modeling. Therefore, we exclude all retweets in our main dataset, and keep the count of retweets in another sub-set (could be used for trend analysis).

Furthermore, we exclude all non-English tweets by checking a "language" tag contained in metadata, and remove tweets with less than three words after regularization. Table 4.1 shows the result of manually screening operated on part of our metadata. Original data amount is 628580154, nearly 13.86% are retweets, and 23.26% are English. After screening, roughly 9.68% tweets were left.

Date	Original	Left	Retweets	English
2018/01	134704400	13643710	18682273	32325983
2018/02	113434467	11301676	18035617	31160072
2018/03	142091427	13124455	18035617	31160072
2018/04	106096293	10144196	13947728	20491484
2018/10	132253567	12634159	18428737	31062896

Table 4.1: Result of Manually filtering on a sample data

Chapter 5

Signal extraction

There are massive topics related to healthcare and diseases and them expand everyday, it's impractical to list them all. [62] defined health-related messages as follows: (1) either a message indicates explicitly the sick (or health problems) of the author; (2) or the message contains the author's worries about health problems (e.g. someone else falling ill, disease outbreak). To surveille diseases, we want to cluster messages(tweets) into different groups, find out what topics are concerned about. The aim of this section is to build a model (or a ensemble of different models) that can: (1) check wether a input document is related to health; (2) assign the checked document to one best match topic. If there is no topic match it, create a new one and assign the document to it (one similar example is hot event discovery). To achieve this, the model should be able to: (1) update the parameters over new input; (2) handle unseen input (such as new words, new topics, new meaning of a seen word, etc.). As far as we know, there is no such model.

In the existing works, there are two types of models: supervised and unsupervised. [5, 14, 35, 72] use supervised machine learning and deep learning techniques to detect flu. Their experimental results show that when used to detect the known disease, supervised models can get high accuracy. However, the performance of such model highly depends on the training set. For topics that are not in the training set, supervised models cannot recognize it. Hence a single end-to-end supervised model (label the data in advance) is not be qualified under this scenario. [45] introduces Ailment Topic Aspect Model (ATAM) that amis to classify aliments. ATAM is an unsupervised model, requires a set of aliments and the prior distribution of them. Its essence is an variant of LDA, and the hyper-parameter K (the number of diseases/topics) is defined before training. Since the aliments and symptoms are pre-defined, ATAM cannot be treated as a pure unsupervised model, and it can't recognize new health event. However, pure unsupervised models cluster data based on its structure, and therefore are not able to identify whether a document is health-related or not. In addition, the generated topics are hard to interpret. Hence, to achieve the requirements of our model, both supervised and unsupervised methods are required.

To extract health-related tweets and topics, [44, 46] described a general framework with 2 main phases: data filtering and topic modeling. The first phase is vital for narrowing down search space. They adopt keyword searching and machine learning as data filtering method. 269 keywords and 20,000 keyphrases were collected and used to filter out irrelevant data roughly. Then 5128 tweets selected randomly from dataset were labeled for training a SVM classifier that identifies wether a input is or isn't health-related. In the second phase, probabilistic topic modeling such as LDA and ATAM are used to cluster tweets and get interpretable topics. [52, 72] followed the same framework to classify tweets in their system, with some variance on data labeling, keywords list and classifier. [21] used exclusion list rather a classifier in phase one.

Before adopting any existing framework, we used Online Biterm Term Modeling (OBTM, a unsupervised Topic modeling algorithm, details can be found in section 5.3.2) on a sample of processed data, to see how many health-related topics can be found. The sample covers all processed tweets created at Jan 2018 (13,643,710 non-retweet english tweets). The algorithm took three days on calculation to cluster documents into 100 topics. Merely few topics contain words related to health while they are hard to interpret. Similar experimental results can be found in [71], where the authors calculate the distribution of topic categories, and find that roughly 5% tweets are about health. To narrow down the search space, decrease the running time and make the final results more interpretable, our project adopts a similar framework with [46]: use the keywords search first, then cluster topics. The details and experimental results are in following sections.

5.1 Keywords search

Influenza is one of the most common diseases and is analyzed most by researchers. For scientific comparison, it was used to test the dataset. [5] used a simple word look-up of "influenza", which may lose massive valuable data. In inspired by [34, 35], we create a list with 26 words highly related to flu based on Flu Symptoms [11] Cambridge Dictionary [10] and relatedwords.org [50]. The complete word list can be seen in table 5.1, note that not all the words from the sources are added to our list. Professional terms are excluded since they are hardly used in colloquialism. Words that are wildly used in other scenarios (such as chill, cold) and phrases that contain keywords in our list (such as asian influenza) are not added. Then tweets are filtered according to the list (ignore case). In this step, we initially adopted a relatively lose filtering strategy: accept tweets containing any string in our list.

Source	Word list
CDC	fever, feverish, sore throat, runny nose, stuffy nose, headache, nasal congestion, diarrhea, bluish lips, bluish face, dehydration
Dictionary	flu, catarrh, cough, common cold, influenza, sniffle, snuffle
relatedwords.org	h1n1, h5n1, coughing, cholera, ebola, epidemic, feverous, measles

Table 5.1: Inclusion list

Table 5.2 shows the number of left tweets after keywords filtering with this list. The test sample are tweets posted in the first 5 days of Oct 2018 (with retweets), and in Jan 2018 (without retweets).

The result corresponds with experiment conducted by [18], where the majority of the filtered tweets are irrelevant to keywords. The possible reasons could be that people will use those words even when they are healthy (such as headache), and some words are substring of

Date	Original	Filtered
2018/10/01	4317376	5984
2018/10/02	4349129	5740
2018/10/03	4417333	5415
2018/10/04	4337327	5676
2018/10/05	4273031	5190
2018/01	134704400	37519

Table 5.2: Tweet counts after flu-related keywords filtering

other words (such as chill-Achill, flu-influence). Another problem found after this step is that the volume of filtered data is far less than expectation. When retweets are included, nearly 0.129% data are left. Exclude retweets, the percentage drops to 0.0279% (although the experiment doesn't operate on a same sample set). In the initial design, geo-tagged tweets (tweets with geographic information) created at certain regions are required. According to [57], geo-tagged tweets are scarce (less than 5% in their experiment), meaning that the percentage of target data could be much lower (less than 0.001%). Under such condition, massive data are required to get a reliable estimation. To get 10000 filtered data, at least 100000000 meta-data are required. The results indicate that, our dataset may be insufficient for analyzing a single disease (such as flu) when the geographic information is required. Therefore, to get a reliable and convincing dataset, we treat all diseases as a whole, expand the inclusion list to more than 9,000 keywords and keyphrases. The sources are given in [44]. Table 5.3 shows the filtering results with new keywords list. More than 30 thousand tweets per day are left, which we believe is sufficient for analysis.

Date	total	average(per day)
2018/01	3590773	115831
2018/02	1087720	40285
2018/03	1108867	35769
2018/04	769700	33465
2018/10	1035587	33406

Table 5.3: Tweet counts after health-related keywords filtering

5.2 Supervised classification

Keywords search helps to screen out the majority of unrelated data. However, as proved in the experiment, it can't guarantee the purity of the rest. To alleviate the content-irrelevance problem found after first round screening, further filtration is required. [21] created another exclusion dictionary containing keywords and phrases indicating tweets should not be included in the filtered dataset, such as "sick and tired". Their experimental result shows such method provides roughly 70% accuracy on their dataset. This method is easy to implement while its accuracy is highly affected by the choice of exclusion words. In section 5.3, we will use unsupervised method to cluster document. Therefore, the outcome of this step decides the final performance of our model. To get a higer accuracy, we decide to use machine learning based classification methods [5]. The model we trained in this step is a binary classifier, aiming to detect health-related tweets. Following are our detailed steps:

1. Data classification: Labeling strategies are various for different uses. [35] labeled their data with three categories, positive, negative and unknown. We adopt a simple binary labeling strategy, since the ambiguity will decrease the performance of unsupervised clustering used in the next section. Document will be labeled 1 if it indeed related to health and can easily be recognized, 0 otherwise. One exception is our news dataset, since all the documents were posted by official healthy news channels (such as BBChealth, CDChealth), we treat them all as positive samples (63279 tweets in total with 59902 unique words). Apart from that, we randomly labeled 50000 negative samples based on the keywords list. 20% data are held out as test set while 20% of training set are used as validation set at each training epoch.

2. Word embeddings: This step aims to transform documents into vectors and create a look-up table. There are generally two transforming methods: (1) randomly initialise word vector for each word and then continuously update them by learning; (2) pre-train word embeddings on training set. The former one is easy to implement but will

need more time on training and it depends more on training set. In our project, we choose to load word embeddings that was pre-trained on large dataset to save training time and increase accuracy. If a word is in the pre-defined dictionary, we will use its vector directly. Otherwise we will initialise its word vector and train it during training. Word2vec [39], Fasttext [30] and Glove [47] are three most common algorithms used to train word embeddings. Although the theories behind them are different, researchers have proven that there is no significant difference among them in practice. While Fasttext and Word2vec train the word vectors through neural networks that are hard to interpret, Glove adopt a more comprehensive method: getting the word representation based on word co-occurrence. For convenience, we use the pre-trained Glove vectors trained on 2 billion tweets with 27 billion tokens and 1.2 million vocabularies. Each word is represented by a 100-dimension vector [47]. An unique index is assigned to each word and its corresponding vector to form up a look-up table. Each document is then transformed to a 2-D matrix based on the table. Note that the number of vocabularies varies among documents. Text can't be reshaped as images, the common solution is pre-define the maximum length of documents and pad missing value with a certain number. In our dataset, most tweets are 5-15 words long, therefore, we define the maximum length of a document is 15. For tweets having less than three words, they are not included in training set. In our implementation, we handle special terms in following rules: For words that either frequently appear in corpus or merely contained by few documents, they are excluded from the document; (2) string '$< pad >'$ is used to pad missing value to extend sentence, and its vector is initialised with zeros; (3) out of vocabulary (OOV) words in new data are represented by string '$< unk >$'. To get meaningful words referring to an extracted topic, we exclude stop words listed by NLTK (The Natural Language Toolkit) [38].

3. Evaluation: We choose binary cross entropy loss as the loss function while training the

model. The formula is:

$$\text{loss}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right)$$

Accuracy and F1 score are used to evaluate the performance of the model, their formulas can be found in section 3.3.5.

4. Modeling and training: The experiment conducted by [72] shows that deep neural network (DNN) outperform traditional machine learning models such as SVM. With the Glove word representation technique [47], their model reaches more than 80% accuracy on their dataset. Based on that, we adopt DNN based models. A basic NLP DNN includes three parts: embedding layer, hidden layers and output layer. Most state-of-the-art models (such as XLNet [70]) adopt transfer learning to improve their performance. Since our focus is not the improvement on supervised algorithm, and our dataset is relatively small, we assume that neural network with simple structure can meet our expectation. TextCNN [31] uses multiple different-size kernels to capture features of documents, with a pooling layer and a fully connected layer. After 20 epochs of training, it gets more than 99% accuracy on training set and more than 98% accuracy on both validation set and test set. The accuracy hardly improves after 30 epochs.

5. Implementation: the major toolkits used for the implementation of our supervised model are Pytorch and Gensim, source code and trained model can be found in our GitHub repository or in the submitted files.

Table 5.4 shows 11 prediction examples of this model. As mentioned before, the accuracy of this model reaches 98.17% on our test data and 98.2% on validation set. The table contains both correct and incorrect predictions. Class 1 represents healthcare-related, 0 otherwise. No.1 to No.6 are correct predictions. However, we mainly care about wrong predictions. Document from No.7 to No.11 are typical incorrect prediction samples. In our training set, word “canada” appears more in negative samples, therefore, sentences with “canada” are more likely be predicted as positive sample. Word “booze” is not in training set, therefore No.8 can’t be correctly recognized. No.9, No.10 and No.11 are noises, and the model indeed correctly classifies them. Based on this observation, the accuracy of this model can be improved by: (1) training on more data; (2) exclude frequent word; (3) using more advanced methods to handle out-of-vocabulary problem (such as unknown word inference).

No.	Text	True class	Prediction
1	the move to digital health care is the future but recruitment and training of quality staff needs to happen al	1	1
2	catholic school less think kids missed point know literally eve	0	0
3	nfl wants players suit over concussions dismissed	1	1
4	ravaged by typhoon philippines faces threat of serious diseases	1	1
5	obama presses leaders to speed ebola response	1	1
6	meat seafood prices rising on drought and disease usda	1	1
7	wrong initio canada arrest america doesn	0	1
8	booze still kills people week	1	0
9	top job opportunity ambulance professionals emt nurses physicians around world	0	1
10	ad feature	1	0
11	course patients first work medical field got stay focused	0	1

Table 5.4: Sample prediction results of Supervised model

5.3 Unsupervised document clustering and topic generation

Supervised models help to screen out documents that are health-related. To detect unseen diseases and healthcare events, we seek solutions on unsupervised algorithm. According to [3], Embeddings Based Clustering and Probability Based Topic Modeling are two common methods used to group documents. This section will introduce both and provide some experimental results of applying them on our dataset. Then we will explain how our model are build upon them. The evaluation criterion used in this section is topic coherence. Since the generated topics are hard to interpret, scoring each topic with a exact value by human judgments could be difficult and objective, and it requires massive labour. Topic coherence helps to evaluate the quality of generated topics. Experimental results have proven that it has positive correlation with human judgments. In our experiment, we adopt the framework proposed by [51] to calculate the coherence, higer value means more coherent. Some libraries such as Gensim provide APIs for using it. For convenience, we uses the online service provided by Palmetto. The Python interface calling the service can be found in our source code. Each generated topic is evaluated by C_a, C_p and their sum together. The test dataset is SearchSnippets, it contains 12,295 documents with 8 clusters and 5,547 unique words. For comparison, the number of cluster in our experiment is the same with that in the test set.

5.3.1 Embeddings based clustering

A simple way for solving clustering problems is using traditional clustering methods. The idea behind it is projecting original data into a vector space, grouping documents based on clustering algorithm (such as K-means). The key point is how to project documents. Traditional methods used to represent documents includes one-hot encoding, bag-of-words and TF-IDF. For example, gievn a corpus with two sentences: (1) “John likes to watch movies. Mary likes movies too.”; (2) “Mary also likes to watch football games”. There are 10 unique words: ‘football’, ‘too’, ‘to’, ‘also’, ‘games’, ‘Mary’, ‘John’, ‘movies’, ‘watch’, ‘likes’. Sentence 1 then can be represented as shown in table 5.5. Equation 5.3 shows the formula of TF-IDF.

	football	too	movies	to	also	games	Mary	John	watch	likes
One-hot	0	1	1	1	0	0	1	1	1	1
BOW	0	1	2	1	0	0	1	1	1	2
TFIDF	0	0.11	0	0	0	0	0	0.11	0	0

Table 5.5: Example of one-hot, bag-of-word and TF-IDF

TF means term frequency while IDF is the abbreviation of Inverse Document Frequency. $n_{d,i}$ is the count of word i in document d, $|D|$ is the count of documents, $|\{j : w_i \in D_d\}|$ means the number of documents that contain word i (1 is added in case of there is no document containing w_i).

$$
\begin{aligned}
TF_{d,i} &= \frac{n_{d,i}}{\sum_w n_{d,i}} \\
IDF_i &= \log \frac{|D|}{1 + |\{j : w_i \in D_d\}|} \\
TFIDF_{d,i} &= TF_{d,i} \cdot IDF_i
\end{aligned}
\tag{5.1}
$$

All of the three traditional methods map words to vectors based on the count of words in the whole corpus and hence has no physical and semantic meaning in a vector space. In addition, the dimensionality of them equals to the number of unique vocabularies in the corpus, hence the vectors projected by them are sparse. Therefore, the distance or cosine among them could contain less or no meaning.

Word embeddings techniques such as Word2vec [39,40] introduce some techniques of projecting words into distributed dense word vectors (as mentioned in section 5.2). Word2vec is an unsupervised machine learning algorithm using neural network, and it can learns relationships between words automatically. Each word is represented by a vector with remarkable linear relationships. One example is: vector("king") - vector("man") + vector("woman") = vector("queen") [38]. Such representation is close to human cognition. Inspired by it, documents, topics are possible to be presented in such way. If words, documents and topics share the same space, then their relationships can be calculated easily. For example, if a document vector is more "close" to a topic vector, it is more likely belongs to that topic. Doc2vec [36] is a variance of Word2vec, it adds one more document vector as the input for each document. Word vectors are shared in corpus while the document is unique for each document. Once the document vectors are trained, clustering algorithms can be applied on them. Here we experiment this method with a simple K-means cluster. After the documents are all grouped, the next step is extracting representive keywords for each topic. We use TF-IDF in our experiment to rank the importance of words.

Table 5.6 shows the experimental results of K-means based topic modeling. Consider the distinguishability of document vectors and the curse of dimensionality problem in K-means, the dimension of document vectors is set to 40. The final score of generated topics is negative 1.62. Since the k-means algorithm is unstable, we made another 5 experiments on the same data. The final average coherence is -1.052 with 0.366 standard deviation. This negative value shows that such simple method generally has no guidance. To get better coherence, more advanced method should be used.

Topic ID	Top_words(10)	C_p	C_a	sum
1	britannica descartes union magician britannica_article encyclopaedia_britannica communism meaning fluid manifesto	-0.372	0.112	-0.260
2	girl episode movie_episode piano tiger magician lyric tiger_wood favorite violin	-0.066	0.099	0.033
3	stanford_edu lecture mit ocw einstein aristotle wolfram maa reasoning opencourseware	-0.473	0.120	-0.354
4	mozilla cisco microprocessor wireless_access client_server mspx zdnet cache ibm sourceforge	-0.421	0.086	-0.335
5	allposters topix sportsline chron cafepress cbs allposters_com showtime boxing trailer	-0.713	0.067	-0.646
6	sewing bull sewing_machine chicago_bull speed_test tiger client_server ticket fund_budget tiger_wood	-0.211	0.064	-0.147
7	economic_development cba medicare referee public_health budget senator union agency internship	-0.329	0.215	-0.114
8	commodity tax fda budget fund_budget union economic_development medicare agency loan	0.034	0.169	0.203
		-2.551	0.931	-1.620

Table 5.6: Experimental results of K-means based clustering

5.3.2 Probability based Topic Modeling

Topic modeling is a typical tool that is frequently used for discovering abstract topics hidden in documents. It mainly uses techniques of Probability. Early progress includes algorithms such as latent semantic indexing (LSI), Unigram language model and Probabilistic latent semantic analysis (PLSA) [6, 27]. Those algorithms abstract document (d), topic (z) and word (w) from corpus, assume that the generation of a document can be considered as picking a sequence of words from dictionary based on certain probabilistic distribution. Latent Dirichlet Allocation (LDA) [8] is a more advanced algorithm under bayesian probability framework. Most of current text modeling algorithms are variants of it. It assumes that: a word is the basic unit of a document, one document could have multiple topics; there are various latent topics that can be characterized by a multinomial distribution over words. The generation of a document can be seen in figure 5.1. It can be explained as following steps:

1. For each document d in corpus D:
   (a) choose the number of words based on poisson distribution and a prior N
   (b) choose the topic distribution θ_d of this document based on dirichlet distribution with prior α
2. For each word w_w in document d:
   (a) choose its topic z_w based on multinomial distribution with prior θ_d
   (b) choose a word w_w based on multinomial probability with prior z_w and β

By training, the word distribution among topics $P(w|z)$ and the topic distribution among documents $P(z|d)$ can be calculated. Then, each document could be assigned to a topic based on its topic distribution. Each topic can be represented by the first few words that are more likely belong to it.

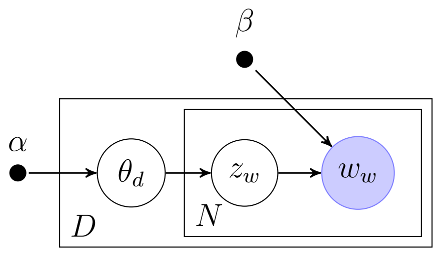


Figure 5.1: Plate Diagram of LDA. [8]

Topic ID	Top_words(10)	C_p	C_a	sum
1	wikipedia football soccer encyclopedia wiki wikipedia_wiki wikipedia_encyclopedia basketball rugby world	-0.000	0.204	0.204
2	system resource business research information political service gov library web	0.335	0.187	0.521
3	sport news tennis com hockey club yahoo volleyball score gymnastics	0.334	0.247	0.581
4	player health school home information journal college page home_page union	-0.136	0.141	0.004
5	edu science research amazon university book culture theory com amazon_com	0.211	0.248	0.459
6	game news sport com movie online democracy play video olympic	0.173	0.202	0.375
7	tournament com online art google search golf photo music degree	0.082	0.204	0.286
8	match party computer republic cancer poker gov democratic president presidential	0.025	0.239	0.264
		1.023	1.671	2.694

Table 5.7: Experimental results of LDA

Table 5.7 shows the experimental result of it. We use the Gensim's implementation, parameters of this model is set to default. After 100 iteration's training, the model gets 2.694 score. Topic 3 gets the higest sum coherence, it can be easily classified to sport related topic by human. Since the parameters of LDA is randomly initialised, each time the output could be different. We made 5 more experiments, and the average sum coherence is 2.64.

In addition to LDA, we experiment with another more advanced topic model, Biterm Topic Model (BTM) [16, 68], which is designed to modeling short text such as tweets. Figure 5.2 demonstrates the generation of corpus in BTM. LDA works well on long document such

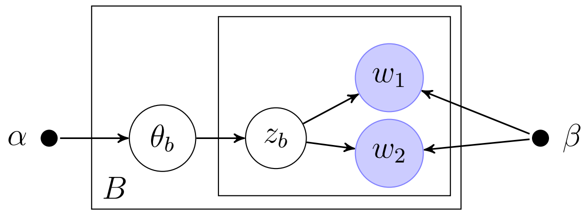


Figure 5.2: Plate Diagram of BTM. [68]

as news, column reports. However, since it captures the word co-occurrence pattern at document-level, its performance may decrease when running on short document such as tweets due to word sparsity problem [68]. BTM provides a feasible solution on short text modeling, it learns the co-occurrence pattern of unordered word pairs (a Biterm) at corpus-level and gets a better retrieval result than LDA and PLSA on short text in the experiment conducted by [68]. The idea behind BTM is that, one words could contain multiple meaning under different scenarios and topics, while a word-pair is more likely belongs to one topic. Documents are then transformed to their corresponding biterms, and the likelihood of the corpus is defined as $P(B|\theta)$ rather than $P(D|\theta)$. Table 5.8 shows the experimental results of BTM applied on SearchSnippets dataset.

Compared with LDA, BTM gets better coherence on both C_p and C_a, its sum score is about 48% higer than that of LDA. 5 more experiments shows that the average sum coherence of

Topic ID	Top_words(10)	C_p	C_a	sum
1	research edu science information school journal university program gov home	0.323	0.221	0.543
2	movie music com art film video news review photo online	0.373	0.277	0.650
3	wikipedia encyclopedia political system wiki wikipedia_wiki culture party theory wikipedia_encyclopedia	0.064	0.167	0.231
4	computer software system web programming memory internet intel com data	0.249	0.223	0.472
5	sport news game football com soccer world match league team	0.445	0.322	0.767
6	business market news stock service trade job finance com information	0.312	0.196	0.508
7	health information cancer disease gov medical drug news calorie healthy	0.479	0.190	0.669
8	amazon com book car amazon_com engine wheel motor war electrical	-0.034	0.190	0.156
		2.211	1.786	3.997

Table 5.8: Experimental results of BTM

BTM is about 4.031 with standard deviation 0.213. In addition, we manually compare the generated topics in each experiment, and find that topics generated by topic modeling with higer sum coherence (more than 0.45) will not change significantly in different experiments (a property of Gibbs sampling), while topics generated by clustering method are different in each experiment.

5.3.3 Our model

Experimental results show that traditional clustering methods can't help to generate interpretable topics. Topic model can find representive words of each topic while grouping documents, it's an ideal solution for our purpose. However, it uses the simple bag of words representation, when handle documents with low frequency words, its interpretability will be affected. To mitigate such problem, an intuitive method is to combine topic model with word embeddings. Lda2vec [42] combines LDA and Word2vec by introducing topic vectors and document vectors. Theoretically, Lda2vec can learn topic, word, document vectors simultaneously while calculating the latent topic distribution, however, in practice we find this model requires massive training time, and the generated topics is hard to understand. Embeded Topic Model (ETM) [20] proposes a novel idea that the word distribution of a topic can be drawn from the Categorical Distribution of the inner production of word embeddings and topic vectors, and it shows how to train the model with back propagation. For our purpose, we need model on short text, hence we take the advantages from both ETM and BTM. The generation of a corpus in our model can be seen as follows, α is the topic matrix with size L $\times$ $|K|$ (L: lengdith of embedding, $|K|$: number of topics), α_k denotes the vector of k_{th} topic. ρ denotes the word embeddings matrix with size L $\times$ $|V|$ ($|V|$: the number of words):

1. For each topic z in topic set Z:
   (a) draw word distribution $\beta_z \sim softmax(\rho^T \alpha_z)$
2. For each biterm b in biterm set B:
   (a) draw a topic assignment $z_b \sim logistc\_normal(\mu, \sigma)$. Equation 5.2 shows the formula of logistc normal. Note that $P(z_{b_i}) = \theta_{b_i}$.

$$\theta_b = softmax(\omega_b); \omega_b \sim Gussian(\mu_b, \sigma_b) \tag{5.2}$$

   (b) draw two words b_{w_1}, $b_{w_2} \sim Cat(\beta_{z,b})$ $(P(b_{w_i} = \beta_{z,b_i}))$

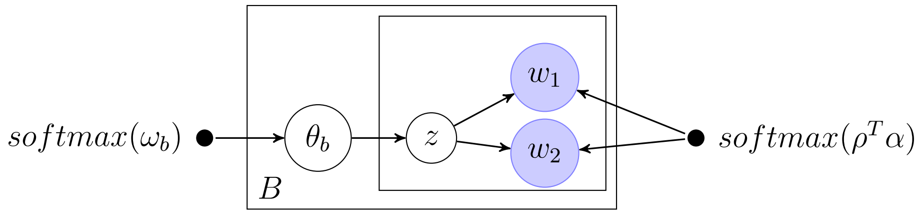


Figure 5.3: Plate Diagram of Our Model.

Choosing logistc normal instead of Dirichlet function as the topic distribution benefits the further parameters inference (parameters of distribution model can be directly optimized by back propagation). Note that μ and σ are two sets of parameters with the size of the topic number $|K|$, ω is a set of $|K|$ numbers. Step 2(a) is the same with BTM while in 2(b), each word of a biterm is drawn from the production of word matrix and topic embeddings. $softmax(\rho^T\alpha_z)$ is a probability vector denotes the probability $P(w|z)$ of each word in the whole corpus. The marginal probability of each biterm under topic matrix α and the likelihood of the whole corpus are defined as follows:

$$\begin{aligned} P(b|\alpha) &= \sum_z P(z)P(b_{w_1}|z,\alpha)P(w_{b_s2}|z,\alpha) \\ &= \sum_\omega P(\omega)P(b_{w_1}|\omega,\alpha)P(w_{b_s2}|\omega,\alpha) \end{aligned} \tag{5.3}$$

$$P(B|\alpha) = \prod_b P(b|\alpha) \tag{5.4}$$

In the model, word embeddings ρ is pre-defined parameters while topic embeddings α is unknown and needed to be trained. To maximize the likelihood is to maximize equation 5.5:

$$\begin{aligned} \log P(B|a) &= \sum_b \sum_z P(z)P(b_{w_1}|z,\alpha)P(b_{w_2}|z,\alpha) \\ &= \sum_b \sum_\omega P(\omega)P(b_{w_1}|\omega,\alpha)P(b_{w_2}|\omega,\alpha) \end{aligned} \tag{5.5}$$

The conditional probability of a word i in biterm b that belongs to topic z can be calculated by equation 5.6 (each biterm belongs to one certain topic):

$$P(b_{w_i}|z,\alpha) = \theta_{b,z}\beta_{z,b_{w_i}} \tag{5.6}$$

$\theta_{b,z}$ is the topic distribution of biterm b, which is generated by the logistc normal in equation 5.2. β_z, b_{w_i} is calculated by $softmax(\rho^T\alpha_k)$ in step 1.

The next question is how to infer the parameters. Equation 5.5 can't be directly optimized since the integral of marginal probability $P(b|\alpha)$ is intractable (EM algorithm cannot be used). In LDA and BTM, Gibbs sampling is used to infer parameters. However, such method requires Dirichlet distribution. Since our model draw words based on the production of $\rho^T\alpha$, same method can't be applied. To infer the parameters, we use Variational inference [67] instead. Variational inference helps to create a simpler distribution that could best approach the original intractable distribution. By adopting variational inference, the original statistical inference problem becomes an optimization problem. Our goal now is to find a distribution $Q(\omega;\phi)$ that has the least "difference" with original marginal distribution of biterm in equation 5.3. Here ω is the variational distribution and ϕ is the variational parameters of q. In step 2(a), we assume that the original topic distribution of a biterm is drawn from a Gussian distribution, therefore, q is defined as a Gussian, whose mean and variance is calculated by its parameters. The next step is learning the variational parameters. [33] proposes a method that could automatically learn the variational parameter based on given input (in our model is the biterm set B), the process can be seen as below:

$$Q(\omega_b;\phi;b) \leftarrow Gussian(\mu_b,\sigma_b) \tag{5.7}$$

$$\mu_b = f(\phi_\mu, b) \tag{5.8}$$

$$\sigma_b = g(\phi_\sigma, b) \tag{5.9}$$

Recall that a biterm is a unordered words pair, and in our model, each word is represented by its embedding. Therefore, in our model, each biterm is defined as the sum of its corresponding words' embeddings (equation 5.10). As stated before, the pre-trained word embeddings has semantic and physic meaning (e.g. vec("phone") + vec("apple") = vec("iphone")), such representation uses this good property. Each biterm is assigned with semantic meaning, and this information will be used in parameter inference.

$$b = b_{w_1} + b_{w_2} \tag{5.10}$$

Function $f(\phi_\mu, b)$ and $g(\phi_\sigma, b)$ are two neural networks with no bias (there is no special requirements of the networks, they could be simple NNs with one linear layer or with more complex structure). They are called “encoder” by [33] since they receive a data (or a batch of data) and output the parameters of its corresponding distribution model (in term of Gussian, the outputs are mean and variance). ϕ_μ and ϕ_σ are the parameters of this two neural networks respectively, and they will be updated by back propagation.

The final step is defining the loss function. Recall that $P(\omega_b)$ is the real topic distribution (un-softmaxed) and $Q(\omega_b; b; \phi)$ is the approximation distribution, ϕ is the parameter set of q, we want to find ϕ that can minimize their “difference” (it's obviously that a simple distribution of $P(\omega_b)$ will benefit the inference of ϕ, which is one reason why we use logistc normal in step 2(a)). The “difference” between $P(\omega_b)$ and $Q(\omega_b; b; \phi)$ is commonly evaluated by the the Kullback-Leibler (KL) divergence [69]. The KL divergence for variational inference is defined as follows:

$$KL(Q(\omega_b; b, \phi)||P(\omega)) = \sum_{\omega} Q(\omega; b, \phi) \log \frac{Q(\omega; b, \phi)}{P(\omega)} \tag{5.11}$$

KL divergence can be treated as information loss, it is always ≥ 0. When KL divergence equals 0, $Q(\omega_b; b, \phi)$ and $P(\omega_b)$ is the same distribution. In our model, we want the KL divergence could be as small as possible. In addition, we want to maximize the log likelihood $\sum_b \log P(b|\omega_b, \alpha)$, and this can be done by maximize equation 5.12:

$$\sum_{b} [\log P(b_{w_1}|\omega_b, \alpha) + \log P(b_{w_2}|\omega_b, \alpha)] \tag{5.12}$$

Therefore, the final loss function is defined as equation 5.13, it tries to maximize the likelihood $\sum_b P(b, \omega_b|\alpha)$:

$$loss(model, B, \rho) = \sum_b [KL(Q(\omega_b; b, \phi_b)||P(\omega_b)) - \log P(b_{w_1}|\omega_b, \alpha) - \log P(b_{w_2}|\omega_b, \alpha)] \tag{5.13}$$

By using the automatic derivation technique, parameters of our model can be updated by back propagation. One problem is that random operation cannot be derived(backpropagated) since it doesn't have gradient. For example, given a variable x,y with arbitrary values, and a variable $z = Gussian(x, y)$, $loss(x, y) = (x - y)^2$, back propagation cannot be applied to the loss function. To solve this problem, make the loss function derivable, we use the reparameterization trick [33]. In terms of the same example, z now equals to $x + Gussian(0, 1) * y$. The distribution of z is the same, while the latter one can be backpropagated. To use the trick, we adopt the logistc normal rather than Dirichlet in step step 2(a). Pseudocode of training the model can be seen in algorithm 1.

The trained model then can be used to infer the topic distribution of documents based on the assumption proposed by [68]:

$$P(z|d) = \sum_{b \in b_d} P(z|b)P(b|d) \tag{5.14}$$

$$P(z|b) = \frac{P(z)P(b_{w_1}|z)P(b_{w_2}|z)}{\sum_z P(z)P(b_{w_1}|z)P(b_{w_2}|z)} \tag{5.15}$$

$$P(b|d) = \frac{n_d(b)}{\sum_b n_d(b)} \tag{5.16}$$

In equation 5.14, the topic distribution $P(z|b)$ of a document is represented by the expectation of the topic proportions of biterms that are extracted from the document. In equation 5.16, $n_d(b)$ represents the number of biterm b in document d. Pseudocode in algorithm 2 explains how this inference works on our model (algorithm 1 and 2 share the same represen-

Algorithm 1: Model training

```
Input: $D$: documents set $D = \{d_1, d_2 \dots d_n\}$
       $\rho$: pre-trained word embeddings
Output: $\alpha$: learned topic embeddings
        $\phi$: learned parameter set of Q
1  Pre-process D, build vocabulary and encode documents in D to indexes
2  Initialise a biterm set $B = \{\}$
3  for $d \in D$ do
4      biterms of $d$ $b_d = extract\_biterm(d)$
5      $B = B \cup b_d$
6  end
7  Initialise topic embeddings $\alpha$ and variational parameters $\phi$ randomly
8  for $i = 1$ to max_iteration do
9      Compute topic distribution of B: $\beta = softmax(\rho^T \alpha)$
10     Split $B$ into smaller batches $B_{batches}$
11     for $X \in B_{batches}$ do
12         for biterm $b \in X$ do
13             Compute $\mu_b = f(\phi_\mu, b)$
14             Compute $\sigma_b = g(\phi_\sigma, b)$
15             Draw $\theta_b \sim logistc\_normal(\phi_\mu, \phi_\sigma)$
16             Compute $P(b_{w_1}|\theta_b) = \theta_b^T \beta_{.,b_{w_1}}$
17             Compute $P(b_{w_2}|\theta_b) = \theta_b^T \beta_{.,b_{w_2}}$
18         end
19     end
20     Compute the loss $l$ based on the loss function
21     if $early\_stop(l)$ then
22         stop iteration
23     end
24     Update parameters $\alpha, \phi$ by back propagation
25 end
```

tation of parameters).

```
Algorithm 2: Topic inference
Input: $d$: a document (encoded)
Output: $P(z|d)$: the topic distribution of d
1  Count the number of each biterm b in document $d$ $n_d(b)$
2  for $b \in B_d$ do
3      Compute $P(b|d) = \frac{n_d(b)}{\sum_b n_d(b)}$
4      Compute $\mu_b = f(\phi_\mu, b)$
5      Compute $\sigma_b = g(\phi_\sigma, b)$
6      Draw $\theta_b \sim logistc\_normal(\mu_b, \sigma_b)$
7  end
8  for topic $z \in Z$ do
9      for $b \in B_d$ do
10         Compute $P(z|b) = \theta_{b_z} \beta_{z,b_{w_1}} \beta_{z,b_{w_2}}$
11         Compute $temp_b = P(z|b)P(b|d)$
12     end
13     Compute $P(z|d) = \sum_b temp_b$
14 end
```

Our model is implemented with Pytorch, topic embeddings α and variational parameters ϕ are represented by three fully connected layers rather than matrixes to make them trainable. There are several basic hyperparameters in our implementation:

1. $|K|$: the number of topics. Same with BTM, $|K|$ is predefined and can't be changed.

2. $|V|$: the number of vocabularies. Vocabularies could be from documents outside the corpus. Larger $|V|$ requires more memory and computation cost while the trained model can infer more documents.

3. $|L|$: the dimensionality of word embeddings.

4. ρ: the pre-trained word embeddings. In our model description, each column represents a word embedding, however, for the convenience, in our implementation, ρ is transposed (ρ^T, with size $|V| \times |L|$). The embeddings can be trained by any algorithms (such as Word2vec, Glove).

5. Hidden size: in our implementation, $f(b, \phi_\mu)$ and $g(\phi_sigma)$ are two fully connected layers whose input size is hidden size and output size is $|K|$. It decides the number of parameters of ϕ to be learned.

6. Activation function: function that adds nonlinearity to $f(b, \phi_\mu)$ and $g(\phi_sigma)$.

7. Batch size: the number of samples fed to the model each time

5.4 System integration and visualization

Both supervised model and

Chapter 6

Potential ethical issues

6.1 Ethical Issues

Although this project got previous ethical approve from campus, it doesn't mean that this project doesn't contain any potential ethical issues. Following are possible ethical issues related to this project.

6.1.1 Privacy Issues

As mentioned above, we will collect social media data from at least one platform in this project. At this stage, a critical ethical issue is that whether our collection will infringe upon others' right. Here "others" are data providers, social media platforms and the users. According to the permutation and combination, this issue can be divided into following six categories:

1. This project and data providers
2. This project and social media platforms
3. This project and users
4. Data provider and social media platforms
5. Data provider and users
6. Social media platforms and users

So far, our data was downloaded from an open source dataset, and in its privacy policy, using the data for research is allowed, there is no confliction between our project and the data provider. However, whether the data providers are authorized by the social media platforms and whether the users allow the platform to share their data are not known (although according to [9], once the data is published on social without requirement for confidentiality, it become public). If neither the platforms nor the providers are authorized by the users, should only they be blamed or our project will also be implicated in the privacy issue (the project didn't collect the data directly from users)?

6.1.2 Trust, Safety and Reliability Issues

In this project, there is actually an basic assumption: the social media data can be analysed for healthcare surveillance. Although [4, 19, 34] showed that social media data can be used to predict the outbreak of disease (means that the congruity between the data and the real records is not a coincidence), but so far, no algorithms can guarantee that their prediction is absolutely correct. If the incorrect results are trusted by the public, they may trigger some trust and reliability issues (such as if the algorithm incorrectly predict an outbreak of disease and the prediction is known by the public, it could results in undesired events such as social panic).

6.1.3 Vulnerable Groups Issues

Suppose that the accuracy of the algorithm is accurate enough. If the prediction is used in positive ways (such as used by hospitals to prepare medical resources for upcoming disease), it can be beneficial. However, if the prediction is used in negative, some people might become vulnerable. (one possible condition is that some people might be isolated because they are from areas where disease transmission could take place).

6.2 Evaluation and Analysis

6.2.1 Privacy Analysis

According to [23], although most social media data are public and are not protected, there are still some private information that can't be collected. In section 6.1.1, six possible sources of ethical issues are listed. The first one comes from social media platforms and their users. Since this project solely analyse Twitter dataset currently, the section mainly focus on it. According to Twitter's privacy policy [64], Twitter can be both public and non-public. While using Twitter, some personal information will be received, such as the type of device users are using and their IP address. Apart from that, users can freely decided whether to share more information such as e-mail address, phone number, a public profile, etc.

For this project, three types of information will be collected from users: (1) The creation time of a tweet; (2) The geographic information showing where the tweet is posted; (3) The content of the tweets. In the section 1.2 "Public Information" of the policy (accessed on Dec 2, 2019), it states that the following activities on Twitter are public: (1) profile information; (2) the language and time zone users' are using; (3) the creation time of users' account; (4) Tweets and its creation time, the version of Twitter etc. In the section 2.1, Twitter states that users can share or hide their location information such as their current precise position or places where they have previously used Twitter, which means that all the location data showing in Twitter are allowed by users (in principle). Based on these statement, users are responsible for the content they post on the Twitter. However,if the platform doesn't obey the policy, collects and uses users' hiden data, discloses those data intentionally or by accident (such as due to technical problem), it should be punished. Taking Facebook as an example, it was fined 5 billion dollars for information leak in July 2019. But note that, sometimes ethical issues could happen even the policy is followed strictly, for example, sharing location or other metadata can actually affect other people than the uploader himself [58]. Platforms need to take such issues in consideration.

The second possible confliction comes from users and third parties (data providers and this project). Firstly, suppose that all the data collected by third parties are authorized by platforms and platforms are authorized by users. In this case, third parties don't need authorization from users, and if they follow the privacy policy, there won't be confliction [61]. But what if the platforms provide unauthorized data to third parties, and third parties use those data in their service. Stated by [23], host social networking sites are responsible for protecting the user data that has been entrusted to them. Application developers can access to data that would not otherwise be available to them through the APIs provided by platforms. This supports that the networking sites should be punished under this condition. The most complex question in this scenario is that whether the third parties should also be punished? John Stuart Mill's Utilitarianism theory [41] states that, wether an act or an intention is morely accessed is depends merely on the its consequence, if a act can maximize the happiness or follow most rules, it is moral. In terms of this project, the aim is to prevent illness and benefit public health, it therefore can increase the happiness of all. But at meanwhile, using unauthorized data breaks the privacy law, which is regarded as immoral based on this theory. In contrast to Utilitarianism, Deontological Theory judge the morality of choices by the will of them insetad of the result, some acts are forbidden if its starting point is morally bad, even if the result is good [2]. Here the using of unauthorized information is immoral through Kantian Reasoning. Assume the act is ok, everyone can use such data without authorization, then there is no unauthorized information, the rule makes no sense and therefore the third parties' keep using those data is immoral.

The thrid possible confliction comes from data providers and this project. In the private policy of Archive site [29], it states that data is free to use but is granted for scholarship and research purposes only. As mentioned above, this project will solely use the data for research analysis, and won't violate the privacy policy.

6.2.2 Reliability Analysis

Unlike the real records collected from hospitals and laboratories, prediction can be incorrect. Following are causes that may lead to prediction failure in this project:

1. Insufficient data: since this algorithm relies solely on social media data, it can be highly affected by the data volume. One supporting example is the experiment done by [21], they found that during festivals, the number of Tweets decreased significantly, and the prediction accuracy also lowered at that time.

2. Noise data: the algorithm uses NLP techniques, it will train a model that can tell whether a Tweet is relevant to healthcare or not. But if there are noises in training dataset, the prediction will be affected. In addition, even if all the train data are correctly labeled, and the model can reach a high accuracy, it still can not guarantee the prediction is what will really happen. For example, if a person is worrying about getting ill, but there are actually no disease outbreak (even human cannot tell), they algorithm may make an wrong prediction. Apart from that, noises can come from unexpected ways, such as a celebrity got ill, and there were massive message talking about that [55].

3. Algorithm itself: so far, there is no model that can simulate the reality with 100% accuracy. Model can reach a high score on certain dataset but may perform badly on others. Especially for new data, the performance of algorithm can not be expected [4].

Assume the algorithm will adopted by hospitals to prepare for preventing against upcoming disease. If the result is false positive (the prediction is positive but wrong), hospitals may waste medical resources, even worse, if the result is revealed, it may causes social panic. If such case happen, who will take the responsibility? According to Birsch's moral responsibility [7], people reaching the three criteria should bear the responsibility: (1) the action of the person caused the harm bring some other bad consequences; (2) the person inflict the harm intentionally or the harm was caused by his recklessness or carelessness; (3) the person know the possible consequences in advance and still choose to neglect it. If we allow the hospitals to use this algorithm without telling them the possible failure, we will be punished. However, if the disadvantages are told in advance, organizations that used the algorithm should take the consequences.

6.2.3 Vulnerable Groups Analysis

This project could harm to some groups if it is used inappropriately. Here the possible vulnerable groups are: (1)citizens living in places that are forecasted a disease spreading, they may become target and at risk of being victims of hate-motivated behaviour; (2) citizens who don't know the forecast, they may lose the equal right when receiving medical resources (people know the prediction in advance have more chances to take actions, such as moving to other city, receive medical treatment early, etc.).

ACM Code of Ethics and Professional Conduct states that we ensure our product won't harm others [1]. Therefore, solutions are required to prevent such condition. One important concern is that who can access to the system and query for the forecast. "I have nothing to hide" argument states that hiding information can be regarded as concealment, it can be benefit criminals rather than common citizens, and therefore it is more harmful to the whole society [59]. According to this statement, the result should be transparent, public should be able to access to the system easily. Centers for Disease Control and Prevention (CDC) adoptes this strategy and update its flu records weekly to public [13]. However, as mentioned before, in a worst case, the prediction can result in social panic, some people even could be isolated. On the contrary, if the system can solely used by one person, it might lose its advantages in disease prevention, since few institutions can take action against the coming illness outbreak. The ideal situation is that the information will be kept solely by healthcare institutions such as official hospitals, and won't be leaked. But in reality, this can hardly be reached, how to balance the social emotion and the prediction usage is the key to solve the problem.

Chapter 7

Reflection and summary

This chapter covers the progress made so far and the remaining tasks needed to be done.

7.1 Project management

This section records the progress of this project so far, starting from Oct 20, 2019. Following is the progress list ranked by chronological order:
Autumn semester (weekly):

- Oct 27, 2019: Searched and read 13 papers related to our project, having a basic idea in mind. Wrote some sketch of functions that will be used in our implementation.

- Nov 03, 2019: Searched and downloaded alternative twitter dataset. Implemented functions that can batch process the downloaded files (unzip the whole directory recursively). Built a preliminary exclusion dictionary used to filter out irrelevant tweets. Implemented functions that can batch filter the unzipped twitter json files. Prepared and past the Gre exam.

- Nov 10, 2019: Searched and collected CDC dataset. Implemented functions that can process the CDC file. Wrote interim report (Data collection section). Searched and read papers related to filtering tweets. Finished coursework 1 of Computer Graphics.

- Nov 17, 2019: Wrote interim report (Data collection and Design section), Implemented functions used to regularize our dataset. Set filtering rules, implemented functions used to label our dataset, and manully labeled 1000 tweets.

- Nov 24, 2019: Found the filtered dataset contains few samples, and can hardly be used to analyze. Therefore, the topic was changed slightly based on the dataset itself. Wrote interim report of Computer Graphics project.
- Dec 01, 2019: Finished coursework2 of Computer Graphics, implemented functions for Computer Graphics project.
- Dec 08, 2019: Finished coursework2 of Computing Ethics, built 3D scene for Computer Graphics project.
- Dec 15, 2019: Finished the Computer Graphics project.

Spring semester: due to Covid-19, the progresses were recorded monthly

- Dec 2019: Rewrote the implementation, fixed some bugs and added more functions for preprocessing
- Jan 2019: Spring festival
- Feb 2019: Change the project design, start to do research on topic model, found more healthcare related dataset.
- Mar 2019: Finished the training of supervised model, experimented with some topic modeling algorithms. Wrote sections about this part.
- Apr 2019: designed and implemented our own model and did experiments on it.

7.2 Conclusion and Future work

Bibliography

[1] ACM. Acm code of ethics and professional conduct. https://www.acm.org/code-of-ethics, 2019. Accessed December 30, 2019.

[2] Alexander, L., and Moore, M. Deontological ethics (2007).

[3] Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., and Kochut, K. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919* (2017).

[4] Andreu-Perez, J., Poon, C. C., Merrifield, R. D., Wong, S. T., and Yang, G.-Z. Big data for health. *IEEE journal of biomedical and health informatics 19*, 4 (2015), 1193–1208.

[5] Aramaki, E., Maskawa, S., and Morita, M. Twitter catches the flu: detecting influenza epidemics using twitter. In *Proceedings of the conference on empirical methods in natural language processing* (2011), Association for Computational Linguistics, pp. 1568–1576.

[6] Baeza-Yates, R., Ribeiro-Neto, B., et al. *Modern information retrieval*, vol. 463. ACM press New York, 1999.

[7] Birsch, D. Moral responsibility for harm caused by computer system failures. *Ethics and Information Technology 6*, 4 (2004), 233–245.

[8] Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *Journal of machine Learning research 3*, Jan (2003), 993–1022.

[9] Burkell, J., Fortier, A., Wong, L. L. Y. C., and Simpson, J. L. Facebook: Public space, or private space? *Information, Communication & Society 17*, 8 (2014), 974–985.

[10] Cambridge Dictionary. https://dictionary.cambridge.org/us/topics/disease-and-illness/colds-and-flu/, 2019. Accessed December 15, 2019.

[11] Centers for Disease Control and Prevention. Flu symptoms & complications. https://www.cdc.gov/flu/symptoms/symptoms.htm, 2019. Accessed December 8, 2019.

[12] Centers for Disease Control and Prevention. U.s. influenza surveillance system: Purpose and methods. https://www.cdc.gov/flu/weekly/overview.htm, 2019. Accessed December 8, 2019.

[13] Centers for Disease Control and Prevention. Weekly u.s. influenza surveillance report. https://www.cdc.gov/flu/weekly/index.htm, 2019. Accessed December 8, 2019.

[14] Chen, M., Hao, Y., Hwang, K., Wang, L., and Wang, L. Disease prediction by machine learning over big data from healthcare communities. *Ieee Access 5* (2017), 8869–8879.

[15] Chen, Y., Crespi, N., Ortiz, A. M., and Shu, L. Reality mining: A prediction algorithm for disease dynamics based on mobile big data. *Information Sciences 379* (2017), 82–93.

[16] Cheng, X., Yan, X., Lan, Y., and Guo, J. Btm: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering 26*, 12 (2014), 2928–2941.

[17] Chollet, F. *Deep Learning with Python*, 1st ed. Manning Publications Co., Greenwich, CT, USA, 2017.

[18] Culotta, A. Towards detecting influenza epidemics by analyzing twitter messages. In *Proceedings of the first workshop on social media analytics* (2010), pp. 115–122.

[19] Denecke, K., and Nejdl, W. How valuable is medical social media data? content analysis of the medical web. *Information Sciences 179*, 12 (2009), 1870–1880.

[20] Dieng, A. B., Ruiz, F. J. R., and Blei, D. M. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907* (2019).

[21] Elkin, L. S., Topal, K., and Bebek, G. Network based model of social media big data predicts contagious disease diffusion. *Information discovery and delivery 45*, 3 (2017), 110–120.

[22] Feldman, R., and Sanger, J. *The text mining handbook: advanced approaches in analyzing unstructured data.* Cambridge university press, 2007.

[23] Felt, A., and Evans, D. Privacy protection for social networking platforms. Citeseer.

[24] Ginsberg, J., Mohebbi, M. H., Patel, R. S., Brammer, L., Smolinski, M. S., and Brilliant, L. Detecting influenza epidemics using search engine query data. *Nature 457*, 7232 (2009), 1012.

[25] Greenwood, S., Perrin, A., and Duggan, M. Social media update 2016. *Pew Research Center 11*, 2 (2016).

[26] Grinstein, U. M. F. G. G., and Wierse, A. *Information visualization in data mining and knowledge discovery.* Morgan Kaufmann, 2002.

[27] Hofmann, T. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (1999), pp. 50–57.

[28] Internet Archive. https://archive.org/, 2019. Accessed December 8, 2019.

[29] Internet Archive. Internet archive's terms of use, privacy policy, and copyright policy. https://archive.org/about/terms.php, 2019. Accessed December 30, 2019.

[30] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[31] Kim, Y. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[32] Kim, Y., Dwivedi, R., Zhang, J., and Jeong, S. R. Competitive intelligence in social media twitter: iphone 6 vs. galaxy s5. *Online Information Review 40*, 1 (2016), 42–61.

[33] Kingma, D. P., and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[34] Lamb, A., Paul, M. J., and Dredze, M. Separating fact from fear: Tracking flu infections on twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2013), pp. 789–795.

[35] Lampos, V., De Bie, T., and Cristianini, N. Flu detector-tracking epidemics on twitter. In *Joint European conference on machine learning and knowledge discovery in databases* (2010), Springer, pp. 599–602.

[36] Le, Q., and Mikolov, T. Distributed representations of sentences and documents. In *International conference on machine learning* (2014), pp. 1188–1196.

[37] Lee, K., Agrawal, A., and Choudhary, A. Real-time disease surveillance using twitter data: demonstration on flu and cancer. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013), ACM, pp. 1474–1477.

[38] Loper, E., and Bird, S. Nltk: The natural language toolkit. *CoRR cs.CL/0205028* (2002).

[39] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[40] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (2013), pp. 3111–3119.

[41] Mill, J. S. Utilitarianism. In *Seven masterpieces of philosophy*. Routledge, 2016, pp. 337–383.

[42] Moody, C. E. Mixing dirichlet topic models and word embeddings to make lda2vec. *arXiv preprint arXiv:1605.02019* (2016).

[43] Omary, Z., and Mtenzi, F. Machine learning approach to identifying the dataset threshold for the performance estimators in supervised learning. *International Journal for Infonomics (IJI) 3*, 3 (2010).

[44] Paul, M. J., and Dredze, M. You are what you tweet: Analyzing twitter for public health. In *Fifth International AAAI Conference on Weblogs and Social Media* (2011).

[45] Paul, M. J., and Dredze, M. A model for mining public health topics from twitter. *Health 11*, 16-16 (2012), 1.

[46] Paul, M. J., and Dredze, M. Discovering health topics in social media using topic models. *PloS one 9*, 8 (2014).

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

[48] Perdana, R. S., and Pinandito, A. Combining likes-retweet analysis and naive bayes classifier within twitter for sentiment analysis. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC) 10*, 1-8 (2018), 41–46.

[49] Qiang, J., Zhenyu, Q., Li, Y., Yuan, Y., and Wu, X. Short text topic modeling techniques, applications, and performance: A survey. *arXiv preprint arXiv:1904.07695* (2019).

[50] relatedwords. Words that are associated with flu. https://relatedwords.org/relatedto/flu, 2019. Accessed December 18, 2019.

[51] Röder, M., Both, A., and Hinneburg, A. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining* (2015), pp. 399–408.

[52] Sadilek, A., Kautz, H., and Silenzio, V. Modeling spread of disease from social interactions. In *Sixth International AAAI Conference on Weblogs and Social Media* (2012).

[53] Sadilek, A., Kautz, H., and Silenzio, V. Predicting disease transmission from geo-tagged micro-blog data. In *Twenty-Sixth AAAI Conference on Artificial Intelligence* (2012).

[54] Salathé, M., and Khandelwal, S. Assessing vaccination sentiments with online social media: implications for infectious disease dynamics and control. *PLoS computational biology 7*, 10 (2011), e1002199.

[55] Schmidt, C. W. Trending now: using social media to predict and track disease outbreaks, 2012.

[56] Signorini, A., Segre, A. M., and Polgreen, P. M. The use of twitter to track levels of disease activity and public concern in the us during the influenza a h1n1 pandemic. *PloS one 6*, 5 (2011), e19467.

[57] Sloan L, M. J. Who tweets with their location? understanding the relationship between demographic characteristics and the use of geoservices and geotagging on twitter. *PloS one 10*, 11 (2015).

[58] Smith, M., Szongott, C., Henne, B., and Von Voigt, G. Big data privacy issues in public social media. In *2012 6th IEEE International Conference on Digital Ecosystems and Technologies (DEST)* (2012), IEEE, pp. 1–6.

[59] Solove, D. J. I've got nothing to hide and other misunderstandings of privacy. *San Diego L. Rev. 44* (2007), 745.

[60] Taehoon Kim and Kevin Wurster. emoji 0.5.4. https://pypi.org/project/emoji/, 2019. Accessed December 15, 2019.

[61] Tankard, C. Big data security. *Network security 2012*, 7 (2012), 5–8.

[62] Tuarob, S., Tucker, C. S., Salathe, M., and Ram, N. Discovering health-related knowledge in social media using ensembles of heterogeneous features. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management* (2013), pp. 1685–1690.

[63] Twitter. Tweet objects. https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json, 2019. Accessed December 17, 2019.

[64] Twitter. Twitter privacy policy. https://twitter.com/en/privacy, 2019. Accessed December 30, 2019.

[65] unicode.org. Full emoji list, v12.1. http://www.unicode.org/emoji/charts/full-emoji-list.html, 2019. Accessed December 15, 2019.

[66] Uysal, A. K., and Gunal, S. The impact of preprocessing on text classification. *Information Processing & Management 50*, 1 (2014), 104–112.

[67] Wainwright, M. J., Jordan, M. I., et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning 1*, 1–2 (2008), 1–305.

[68] Yan, X., Guo, J., Lan, Y., and Cheng, X. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web* (2013), pp. 1445–1456.

[69] Yang, X. Understanding the variational lower bound, 2017.

[70] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems* (2019), pp. 5754–5764.

[71] Zhao, W. X., Jiang, J., Weng, J., He, J., Lim, E.-P., Yan, H., and Li, X. Comparing twitter and traditional media using topic models. In *European conference on information retrieval* (2011), Springer, pp. 338–349.

[72] Șerban, O., Thapen, N., Maginnis, B., Hankin, C., and Foot, V. Real-time processing of social media with sentinel: a syndromic surveillance system incorporating deep learning for health classification. *Information Processing & Management 56*, 3 (2019), 1166–1184.