Tales of a Coronavirus Pandemic: Topic Modelling with Short-Text Data

BY

Adam Shen

A THESIS SUBMITTED TO

The Faculty of Graduate and Postdoctoral Affairs

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

IN

Statistics

Carleton University

Ottawa, Ontario, Canada

© 2021

Adam Shen

Master of Science (2021)
Mathematics and Statistics

Carleton University
Ottawa, Ontario, Canada

TITLE: Tales of a Coronavirus Pandemic:
Topic Modelling with Short-Text Data

AUTHOR: Adam Shen
B.Sc., (Statistics)
McMaster University
Hamilton, Ontario, Canada

SUPERVISORS: Dr. David Campbell
Dr. Song Cai
Dr. Shirley Mills

NUMBER OF PAGES: xiii, 72

Abstract

With more than 13 million tweets collected spanning between March 2020 to November 2020 relating to the COVID-19 global pandemic, the topics of discussion are investigated using topic models – statistical models that learn latent topics present in a collection of documents.

Topic modelling is first conducted using Latent Dirichlet Allocation (LDA), a method that has seen great success when applied to formal texts. As LDA attempts to learn latent topics by analysing term co-occurrences within documents, it can encounter difficulties in the learning process when presented with shorter documents such as tweets. To address the inadequacies of LDA applied to short-text, a second topic modelling technique is considered, known as the Biterm Topic Model (BTM), which instead analyses term co-occurrences over the entire collection of documents.

Comparing the performances of LDA and BTM, it was found that the topic quality of BTM was superior to that of LDA.

Acknowledgements

I would like to thank Dr. David Campbell for his patience, positivity, and encouragement throughout the duration of this thesis. I have learned so much from him in such a short amount of time and I am extremely grateful for the many opportunities he has given me.

I would not have made it this far without the kindness, guidance, and support from Dr. Song Cai, especially in my first semester when I was struggling at just about everything. In addition, this thesis would not have been possible without his computing server which I had the privilege of using all to myself.

Thank you to Dr. Shirley Mills, MITACS, and Carleton University for funding the project upon which this thesis was built.

Contents

Abstract	iii
Acknowledgements	iv
List of Figures	viii
List of Tables	xii
List of Algorithms	xiii
1 Introduction	1
1.1 The coronavirus disease 2019 pandemic	1
1.2 About Twitter	2
1.2.1 Terminology	2
1.2.2 Increasing one’s own public presence	3
1.3 Thesis outline	3
2 Data preparation	5
2.1 Data collection	5
2.2 Data verification	6
2.3 Text cleaning	7
2.4 Language detection	7

2.5	Quality check	9
3	Exploratory analysis	10
3.1	Results	10
3.2	Discussion	17
4	Sentiment analysis	20
4.1	Background	20
4.2	How <code>sentimentr</code> works	21
4.3	Methods	23
4.4	Results	24
4.5	Discussion	28
5	Topic modelling	29
5.1	Introduction	29
5.2	Latent Dirichlet Allocation (LDA)	30
5.2.1	Model outline	31
5.2.2	Parameter estimation	33
5.2.3	Perplexity	35
5.3	Biterm Topic Model (BTM)	36
5.3.1	Model outline	37
5.3.2	Parameter estimation	40
5.3.3	Document-topic proportions	41
5.3.4	Log-likelihood	42
5.4	Comparison of LDA and BTM models	42
5.5	Methods	43
5.5.1	LDA	43
5.5.2	BTM	45
5.6	Results	46

5.6.1	LDA	46
5.6.2	BTM	52
5.6.3	Topic quality	60
5.7	Discussion	61
6	Conclusions	63
	Bibliography	65

List of Figures

3.1	Number of tweets (thousands) available per day.	11
3.2	Number of tweets (millions) that used between 0 and 15 hashtags. . .	12
3.3	Top 30 hashtags and their frequencies (thousands), excluding hashtags related to the search queries.	13
3.4	Top three monthly hashtags and their frequencies (thousands), excluding hashtags related to the search queries. Horizontal scales differ across months due to differing amounts of tweets available for each month.	14
3.5	Wordcloud of the most used terms in the collected tweets, excluding stopwords and terms related to the search queries. Increased text size and darker colour corresponds to increased usage.	15
3.6	Wordclouds of terms co-occurring with the captioned term in a tweet, excluding stopwords, terms related to the search query, and the captioned term. Increased text size and darker colour corresponds to increased usage.	16
4.1	Monthly average sentiments for tweets, using the Jockers-Rinker and NRC lexicons. Point sizes represent the proportion of tweets contributed by the given month, relative to other months.	25

4.2	Sentiment scores for the five lowest scoring tweets of March 2020, using the Jockers-Rinker and NRC lexicons. Text shown is the cleaned text, i.e. removal of mentions, symbols, emojis, and demotion of hashtags.	26
4.3	Sentiment scores for the five lowest scoring tweets of October 2020, using the Jockers-Rinker and NRC lexicons. Text shown is the cleaned text, i.e. removal of mentions, symbols, emojis, and demotion of hashtags.	27
5.1	Graphical model representation of LDA. Nodes in the graph represent random variables; shaded nodes are observed variables. Plates denote replication, with the number of replicates given in the bottom right corner of the plate.	32
5.2	Graphical model representation of BTM. Nodes in the graph represent random variables; shaded nodes are observed variables. Plates denote replication, with the number of replicates given in the bottom right corner of the plate.	39
5.3	Perplexities for the nine models trained in the first tuning step, evaluated on the validation set. K , the number of topics, was fixed while varying values of α and β . The hyperparameters corresponding to the model with the lowest perplexity were $\alpha = 100/K$ and $\beta = 0.1$	47
5.4	Perplexities for six of the seven models trained in the second tuning step, plus the best model from the first tuning step, evaluated on the validation set. α and β were fixed, while varying over values of K . Perplexities involving the $K = 500$ model were unable to be calculated due to memory constraints.	48
5.5	Monthly document-topic allocations scaled by the number of tweets available per month. The five topics highlighted attained the highest scaled allocation values overall.	49

5.6	Monthly top three topics by scaled allocations.	50
5.7	Top 20 most probable terms for topics appearing in Figure 5.5 and Figure 5.6	51
5.8	Log-likelihoods for the nine models trained in the first tuning step, evaluated on the validation set. K , the number of topics, was fixed at 10 while varying values of α and β . The hyperparameters corresponding to the model with the highest log-likelihood were $\alpha = 50/K$ and $\beta = 0.1$	52
5.9	Log-likelihoods for the seven models trained in the second tuning step, plus the best model from the first tuning step, evaluated on the validation set (black) and the testing set (green). α and β were fixed, while varying over values of K	53
5.10	Monthly document-topic allocations scaled by the number of tweets available per month. The five topics highlighted obtained the highest scaled allocation values overall.	54
5.11	Monthly top three topics by scaled allocations.	56
5.12	Biterm clusters for topics appearing in Figure 5.10 and Figure 5.11 . Each cluster is an undirected graph as biterms are unordered pairs of terms. Increased node (term) size corresponds to higher topic-term probability. Increased thickness and darkness of edges (links) corresponds to higher co-occurrences within the topic. As biterms were computed using a window size of 15, adjacent nodes may not have necessarily appeared adjacently in the original text.	58
5.13	Top 20 most probable terms for topics appearing in Figure 5.10 and Figure 5.11	59

5.14	Mean coherence scores and 95% t -intervals for the 500 topics of the BTM and LDA models using the top $M = \{5, 10, 15, 20\}$ terms of each topic, evaluated on the testing data. A higher coherence score means that a topic is more coherent. Using a two-sample t -test, the mean coherence of topics learned by BTM was found to be significantly ($p < 0.001$) greater than the mean coherence of topics learned by LDA for the each of the four values of M considered here.	60
------	--	----

List of Tables

2.1 A sample of the raw data obtained from the Twitter API using the
rtweet¹³ package. 8

List of Algorithms

5.1	Collapsed Gibbs sampling algorithm for LDA.	35
5.2	Collapsed Gibbs sampling algorithm for BTM.	40

Chapter 1

Introduction

1.1 The coronavirus disease 2019 pandemic

The coronavirus disease 2019 (COVID-19) pandemic is an ongoing pandemic caused by a strain of coronavirus known as severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2)¹. This strain of coronavirus was first identified in Wuhan, China in December of 2019¹. At the time of writing, more than one year has elapsed since the documentation of the initial case. In this time, more than 86.5 million cases have been confirmed globally, of which, more than 1.87 million cases have resulted in death².

Transmission of COVID-19 can occur between humans when a person comes into contact with the respiratory droplets resulting from the coughing, sneezing, talking, or breathing of an infected person¹. Symptoms of COVID-19 can vary greatly from person to person, with typical symptoms including, but not limited to: fever, cough, fatigue, difficulty breathing, lack of smell, and lack of taste¹.

The effect of the pandemic has been felt on a global scale through the economy³, travel restrictions⁴, unemployment⁵, supply shortages⁶, heightened racism and xenophobia towards people of Chinese and East Asian descent^{7,8}, and protests for change

in healthcare and political systems^{9,10}.

Mass hysteria was bound to ensue in the early days of the pandemic due to uncertainty surrounding the authenticity of information sources, and a cult-like uprising of deniers of scientific evidence. In times like these where the general public has been advised to stay home except for essential purposes and in-person social gatherings have been restricted, one way to stay connected with friends and family, get the latest news, and discuss current events is through social networking platforms.

1.2 About Twitter

Twitter is an American social networking platform with approximately 321 million active users (as of February 2019)¹¹ across the world. Twitter is available in most countries and can be accessed from a web browser or a mobile application. The platform allows users to publish posts called *tweets*, consisting of text up to 280 characters, as well as media including photos, GIFs, and videos. For the average user, published tweets are visible to the public (including those without Twitter accounts). As a result of the character limit, users may resort to slang or abbreviations to convey their ideas. After a tweet has been published (publicly), the original poster, as well as other Twitter users, can interact with the tweet in three ways: *like*, *comment*, or *retweet* (re-post a tweet to one's own collection of tweets).

1.2.1 Terminology

- Hashtag: a keyword or group of words (without spaces or punctuation) preceded by a # symbol. Hashtags can act as a topic identifier in a tweet.
- Mention: the tagging of one or more Twitter users by including their Twitter handles (usernames) preceded by a @ symbol. Mentions can be used to direct a

tweet at one or more Twitter users and notify them of the tweet.

- Follow: keep up with a Twitter user's activity by displaying their tweets, likes, comments, and retweets on your main feed.

1.2.2 Increasing one's own public presence

While hashtags are not mandatory, hashtags can be clicked on to view all other tweets sharing the same hashtag. Through Twitter's *explore* feature, one can view trending keywords/hashtags and associated tweets. Outside of the overall trending topics, the explore feature can be filtered to include tweets on news, politics, and entertainment. Therefore, Twitter users can garner more attention towards their tweets by taking part in the discussion of current events.

Taking advantage of the fact that the users that one follows dictates the content shown on the main feed, having users interact with your tweets (like, comment, or retweet) allows your content to travel through their social networks, appearing as activity for their followers. With a moderately-sized number of followers, information (whether correct or not) can travel extremely quickly across users' social networks.

1.3 Thesis outline

Twitter users have voiced many differing opinions on the pandemic, its effect on our daily lives, its handling by government organizations, and its intertwining with global affairs varying from region to region. While Twitter's application programming interface (API) allows for the easy access and retrieval of tweet data, the amount of available data (despite only being a sample) is enormous and would be overwhelming for manual analysis by humans.

The purpose of this thesis is to employ modern information retrieval techniques,

namely Latent Dirichlet Allocation and Biterm Topic Modelling, in order to summarise the obtained pandemic tweet data by their latent topics. With this information, an analysis of the topical trends can be performed to elucidate the most prominent topics of discussion, as well as how the topics of discussion vary over the course of the pandemic.

The processing of the raw text data preceding any analyses is detailed in Chapter 2. In Chapter 3, an exploratory analysis is performed to give a basic overview of the data and illustrate some of the insights that can be drawn from text data without the use of models. Chapter 4 is an extension of the exploratory analysis: a sentiment analysis is performed using an augmented dictionary lookup – one that considers parts of speech known as [valence shifters](#). In Chapter 5, topic modelling with the classical method of Latent Dirichlet Allocation (LDA), and its short-text variant, the Biterm Topic Model (BTM), is performed. Finally, Chapter 6 discusses the shortcomings of LDA and BTM and a novel approach to topic modelling that addresses some of the shortcomings of LDA and BTM.

All computational methods in this thesis were performed in **R** 4.0^{[12](#)} with the aid of numerous open-source packages^{[13–40](#)}.

Chapter 2

Data preparation

2.1 Data collection

Twitter data can be obtained from the Twitter Application Programming Interface (API) by specifying keywords that the returned data should contain, either in plain text or hashtag. Data from up to nine days prior to the time of the call to the Twitter API can be retrieved. The rate limit allows a maximum retrieval of 18,000 observations every 15 minutes. For each successful query, a simple random sample of data containing the specified keywords is returned. Individual calls to the Twitter API are independent and as a result, duplicate data may be obtained between calls.

The keywords used in the queries to obtain the data for this thesis included:

covid	covid19	covid_19
coronavirus	epidemic	pandemic
social distancing	socialdistancing	lockdown

Twitter data were obtained at the rate limit continuously between March 27, 2020 and November 14, 2020. After removing duplicate data, a total of 61,001,110 unique observations remained consisting of 14,458,259 (23.7%) tweets and 46,542,851 (76.3%)

retweets.

Each observation had metadata fields including:

- `user_id` : a unique numeric identifier for the user performing the tweet/retweet
- `status_id` : a unique numeric identifier for the tweet/retweet
- `created_at` : the date and time at which the tweet/retweet was performed (in UTC time)
- `is_retweet` : a logical value indicating whether the observation was a retweet
- `hashtags` : a vector of hashtags contained within the tweet/retweet
- `lang` : a character value indicating the language that Twitter has detected

Other metadata fields such as city, country, and geographical coordinates were also included. However, since the availability of data for these fields was highly dependent on the settings of the individual users, many of these fields were mostly missing values and were therefore unusable. A sample of the raw data can be found in [Table 2.1](#).

All text-related analyses were conducted using only tweets, i.e. retweets were excluded, as tweets were most representative of the ideas of individual users. Henceforth, the term “data” will refer to tweets only, unless otherwise specified. The word “term” will be used when words and hashtags are referred to without distinction.

2.2 Data verification

It was found that there were instances of tweets and retweets that had been collected despite not matching any of the search queries. Therefore, another pass through of the data was done to ensure tweets contained at least one of the queried terms. Of the 14,458,259 collected tweets, 681,654 (4.7%) tweets did not contain any of the

queried terms and were removed, leaving 13,776,605 tweets containing one or more of the queried terms.

2.3 Text cleaning

The major steps of the text cleaning included:

- Conversion of the text encoding from **UTF-8** to **ASCII**, thereby removing any symbols, emojis, and characters that were not native to the English alphabet. This was done for compatibility across machines and packages.
- Conversion of all text to lower case.
- Removal of user mentions and links.
- Demotion of hashtags to words (by removing the **#** sign). This was done due to the inconsistent nature of how Twitter users incorporate hashtags into their tweets – some users replace keywords mid-sentence with hashtags, some accumulate all relevant hashtags at the end of the tweet, and some use a combination of both.

Additional text cleaning occurred in chapters 3 and 5, after further exploration.

2.4 Language detection

While Twitter automatically classifies the language of tweets using its own algorithms and allows for retrieval of tweets matching specific languages, it is often possible that the retrieved tweets can contain a mixture of languages. For the results of the proceeding exploratory analysis to be interpretable, in combination with the knowledge of the role valence shifters play in the English language, only tweets that were either *mostly* or entirely in English were retained.

user_id	status_id	created_at	text	is_retweet	hashtags	lang
833499844517961728	1255290686250811392	2020-04-29 00:19:56	The NY Dem Presidential Primary on 6/23 is canceled due to COVID-19. Other elections on 6/23 will still take place: https://t.co/0P6PMPq20h - Rock the Vote	FALSE	NA	en
581180392	1254511313331597314	2020-04-26 20:42:59	In the midst of the current #pandemic, it is more important than ever to get creative with your #homeworkout. One great #trainfromhome option includes making a sandbag from a laundry bag. Here's how to implement sandbag circuits: https://t.co/Ottxy2YZAq	FALSE	"pandemic", "homeworkout", "trainfromhome"	en
819296296062201857	1304087649011953665	2020-09-10 16:01:38	#Masks serve 2 purposes: 1) To remind everyone that there is supposed to be a deadly pandemic, despite nobody knowing anybody who is sick with covid, and 2) To increase incidence of pulmonary illnesses, which will later be blamed on covid. Proof: See my pinned tweet.	FALSE	"Masks"	en
2350488460	1322387795718119425	2020-10-31 03:59:53	#GOPBetrayedAmerica decisions are being made to take your voice/vote away. The same #GOPCorruptionOverCountry that won't respond with help for every American during this once a century pandemic https://t.co/Gg5gecx7tb	FALSE	"GOPBetrayedAmerica", "GOPCorruptionOverCountry"	en

Table 2.1: A sample of the raw data obtained from the Twitter API using the `rtweet` ¹³ package.

The **R** packages `cld2`²⁴ (a wrapper for the **C++** implementation of Google’s Compact Language Detector 2) and `cld3`²⁵ (a wrapper for the **C++** implementation of Google’s Compact Language Detector 3) were used to detect the language of each tweet. Unfortunately, the results of `cld3`’s language detection were very poor even for tweets that were truly entirely in English. This was possibly due to the short nature of tweets, combined with the uncertainty surrounding scientific terms such as COVID-19. On the other hand, the results of `cld2` were very promising. As such, only tweets that were classified as English by both Twitter and `cld2` were kept.

A total of 192,233 (1.4%) tweets were removed, with 13,584,372 tweets remaining.

2.5 Quality check

Tweets that had fewer than two terms were removed. Upon further inspection into the number of hashtags used in the collected tweets, it was found that tweets that contained an excessive number of hashtags had no semantic value and were either spam or to gain attention by mentioning as many trending topics as possible. Therefore, tweets containing more than 15 hashtags were removed.

A total of 14,751 (0.1%) tweets were removed, leaving a final total of 13,569,621 tweets.

Chapter 3

Exploratory analysis

3.1 Results

Considering that the `rtweet`¹³ package can collect tweets up to nine days prior to the current time of collection, the creation timestamps of the collected tweets were investigated to ensure that the data was suitable for further analysis. Collected tweets were dated between the period of March 27, 2020 and November 14, 2020. Within this interval of 232 days, 51 (22%) days had no tweet data due to software failures in the collection code. Counts of the number of tweets available per day are given in [Figure 3.1](#).

Of the data remaining after removing tweets that contained more than 15 hashtags, it was seen that the vast majority of tweets did not use any hashtags ([Figure 3.2](#)). For tweets that did contain hashtags, the most common hashtags were those that matched the search queries, as one would expect. Excluding hashtags that matched the search queries, the 30 most popular hashtags are displayed in [Figure 3.3](#).

Throughout the duration of the collected tweets, there were many variants of hashtags advising others to stay home. This included hashtags such as `stayhome`,

Figure 3.1: Number of tweets (thousands) available per day.

`stayathome`, `stayhomestaysafe`, `stayhomesavelives`, and `stayhome[***]` where `[***]` was a city, state, province, or country. As such, hashtags containing the word `stay` and any of `home`, `safe`, `save`, `lives`, or `life` were collapsed into the hashtag `stay*`. The `stay*` variant hashtags were the most popular, with 95,538 occurrences out of 5,316,541 (1.8%) non-query hashtags. The second most popular hashtag was `trump`, with related hashtags `trumpvirus` at rank 12, and `maga` (Make America Great Again) at rank 16. Lastly, many of the top hashtags referred to geographical locations including China, India, Australia, USA, Canada, and UK.

Due to the large amount of variability in the number of tweets obtained for each day (Figure 3.1), with many days with missing data, the top three non-query hashtags for each month were sought (Figure 3.4), rather than the daily top hashtags. The all-time top hashtag, `stay*`, persisted in the top three most used hashtags for all months, only losing its first place position in the months of September, October, and

Figure 3.2: Number of tweets (millions) that used between 0 and 15 hashtags.

November. Interestingly, during the months of March, April, and May, the usage of **stay*** was substantially greater than other top hashtags. Over the months of June, July, and August, while **stay*** remained the top hashtag, its usage was eventually equalized with the other top hashtags. From September onwards, **stay*** consistently remained at position three.

The **trump** hashtag appeared in the top three hashtags for the months of March to June, and reappeared in the top three hashtags in the months of October and November. In September, the top two hashtags, **protectenhypen** and **respect_enhypen**, were in reference to the Korean boy-band Enhypen, whose members were swarmed by fans at a Seoul airport due to the leakage of their private flight schedule⁴¹.

To visualize term frequencies relative to each other, wordclouds were constructed with increased text size and darker colour corresponding to increased usage. Term frequencies were counted following the removal of terms related to the search queries as

Figure 3.3: Top 30 hashtags and their frequencies (thousands), excluding hashtags related to the search queries.

well as the removal of stopwords. Stopwords were removed according to the `snowball` stopword lexicon²².

A general wordcloud (Figure 3.5) was constructed for an overview of the terms contained in the data. The most used term was `people` with a total of 1,753,528 usages. `trump` was the seventh most used term, with 841,979 usages. Unfortunately, a global wordcloud may not offer too much information due to a lack of focus, as evidenced by the dark green centre with surrounding words being much lighter.

It was therefore of greater interest to construct wordclouds for terms co-occurring in tweets that mentioned the (exact) terms: `trump`, `china`, `election2020`, `economy`, `health`, and `vaccine` (Figure 3.6). These wordclouds were created similar to the general wordcloud: term frequencies were counted following the removal of terms related to the search queries and removal of stopwords (`snowball` lexicon²²), followed

Figure 3.4: Top three monthly hashtags and their frequencies (thousands), excluding hashtags related to the search queries. Horizontal scales differ across months due to differing amounts of tweets available for each month.

by the additional removal of the term of interest.

From [Figure 3.6](#), it was seen that the term **trump** appeared in all wordclouds (excluding its own). In addition, **trump** was the central term in the **china**, **election2020**, and **economy** wordclouds, while still remaining close to the centre in the **health** and **vaccine** wordclouds. Lastly, it was observed that tweets that mentioned **vaccine**, **trump**, **health**, and **economy** were the most focused (smaller vocabulary size for terms appearing more than once) as evidenced by the more abundant increased-size and dark green terms spanning towards the outer edges of the wordclouds, with the **vaccine** wordcloud being the most focused.

3.2 Discussion

From [Figure 3.1](#), it was seen that many dates had no available tweet data. For the dates that did have tweet data, there was a large amount of variability in the number of tweets collected. This showcases one of the difficulties of obtaining data from the Twitter API and can be explained by a variety of factors:

1. *Rate limit and data window.* Due to the fact that Twitter data obtained would have been created up to nine days prior to the time of query, for a topic as popular as the COVID-19 pandemic, it is almost certain that there existed an incredibly large amount of tweet and retweet data related to the search terms. With a rate limit (maximum number of data points retrieved in one call) of 18,000, presumably much smaller than the total number of tweets and retweets related to the search terms, one would require numerous rounds of collections to obtain a sizeable fraction of the existing data within this moving nine day window. In addition, since each collection is performed independently from one another, it is possible to obtain duplicated data, and on rare occasions unwanted data ([Section 2.2](#)).
2. *Collection of retweet data.* As the original intention behind the collection of this particular COVID-19 Twitter data was not for this thesis, retweets were included in the collection step. As the call to the Twitter API allows for the exclusion of retweets, enabling it would have been desirable. As seen in [Section 2.1](#), 76.3% of the original collected data consisted of retweets and were inevitably excluded from the analyses performed in this thesis.

Had metadata fields that detailed the geographic origin of the collected tweet data been available, an additional analysis of the topics of discussion by region would have been of interest. Unfortunately, as explained in [Section 2.1](#), the majority of values for these fields were missing. Although a sizeable number of tweets mentioned

geographical regions in hashtag form (Figure 3.3), and quite likely, an even larger number of tweets mentioned geographical regions in plain text (as hashtags are not mandatory), it would not have been correct to assume the geographic origin of the tweet based on the region of discussion. For example, tweets that discussed the origin of COVID-19 would have mentioned Wuhan or China, but this would not have offered any geographical information on the user. In addition, as Twitter is a loosely moderated platform, it is acceptable to share opinions on matters one knows nothing about.

The appearances of the term `trump` in Figure 3.5 and Figure 3.6 suggest that discussions of Donald Trump, the former president of the United States, are heavily intertwined with many of the sub-topics of these COVID-19 tweets. One should recall and take into consideration that the term frequencies in the `trump` wordcloud of Figure 3.6 were counted from tweets that explicitly mentioned `trump`. As such, the term frequencies of tweets that talked about Trump without mentioning his name were not counted. This includes:

- Tweets where users referred to Trump by his Twitter handle (recall that user mentions were removed as part of the cleaning process; Section 2.3) and did not mention his name again for the remainder of the tweet
- Tweets where Trump was referred to by another title/nickname such as `president` or `POTUS`
- Tweets directed at Trump's family members, referring to Trump by a familial relation such as `your father`

Finally, it is worth noting that when working with informal text, tallied term frequencies are estimates. Due to the presence of hashtags, character limits, and to a lesser extent, misspellings, these estimates are likely to be underestimated. For

example, a hashtag such as `nomasknoservice` would not generate counts for `no`, `mask`, and `service`, but would instead generate a count for the possibly unique term, `nomasknoservice`. At this time there does not exist a simple solution to split hashtags accurately into their component terms, especially for data of this size that uses both colloquial and modern scientific terms.

Chapter 4

Sentiment analysis

4.1 Background

Sentiment analysis aims to classify the polarity of text as either negative or positive, based on the sign of the final computed sentiment score. Basic sentiment analysis methods involve a dictionary lookup: words in a document are compared against words in a polarity lexicon, where each polarized word has a pre-determined score, and the final sentiment score is computed as the sum of the individual polarized words' scores.

The disadvantage of simple dictionary lookups is that they do not consider parts of speech that can affect polarized words. These parts of speech are known as *valence shifters* and can be broken down into four categories³²:

- Negators: flips the sign of a polarized word (e.g., I am **not** *sick*.)
- Amplifiers: increases the impact of a polarized word (e.g., I am **definitely** *sick*.)
- De-amplifiers: reduces the impact of a polarized word (e.g., I am **somewhat** *sick*.)

- Adversative conjunctions: overrules the previous clause containing a polarized word (e.g., I was *sick* **but** I feel better now.)

Without consideration for valence shifters, a basic sentiment analysis may not accurately capture the intended sentiment of the given text. The **R** package `sentimentr`³² attempts to address this shortcoming by employing an augmented dictionary lookup, which combines a basic dictionary lookup of polarized terms with consideration for valence shifters.

4.2 How `sentimentr` works

Consider a sentence, s_i , as an ordered bag of words such that

$$s_i = \{w_{i,1}, \dots, w_{i,n}\},$$

where $w_{i,j}$ represents the j^{th} word of the i^{th} sentence. All punctuation is stripped from the sentence with the exceptions of commas, colons, and semicolons, which will be denoted as w^c and referred to as “comma-words”. Polarized words, $w_{i,j}^p$, of the sentence are tagged as positive or negative (according to the chosen sentiment lexicon).

After the tagging of polarized words, context clusters are formed about each polarized word. Cluster boundaries take into consideration the locations of comma-words, as comma-words can be thought of as possible “changes in thought”. Therefore a context cluster, $c_{i,j}$, centred at $w_{i,j}^p$ with lower bound, L , and upper bound U , can be written as an ordered sequence of words

$$c_{i,j} = \{L, \dots, w_{i,j}^p, \dots, U\}$$

where:

- $L = \max \{w_{i,j-n_{\text{before}}}^p, w_{i,1}, \max \{w_{i,j}^c < w_{i,j}^p\}\}$
- $U = \min \{w_{i,j+n_{\text{after}}}^p, w_{i,n}, \min \{w_{i,j}^c > w_{i,j}^p\}\}$
- $j - n_{\text{before}}$ is the position of the word n_{before} units before the polarized word, and $j + n_{\text{after}}$ is the position of the word n_{after} units after the polarized word. By default, $n_{\text{before}} = 4$ and $n_{\text{after}} = 2$.

Informally, the lower bound, L , is the closest of the most recent comma-word before the polarized centre word, the word n_{before} units before the polarized word, or the first word in the sentence should $j - n_{\text{before}}$ go out of range, to the polarized centre word. Similarly, the upper bound, U , is the closest of the most recent comma-word after the polarized centre word, the word n_{after} units after the polarized word, or the last word of the sentence should $j + n_{\text{after}}$ go out of range, to the polarized centre word.

It is within this context cluster that valence shifters are considered and affect the weighting of the polarized centre word.

- Amplifiers, $w_{i,j}^a$, increase the polarity of the context cluster centre while de-amplifiers, $w_{i,j}^d$ decrease the polarity of the context cluster centre.
- Negators, $w_{i,j}^{\text{neg}}$, flip the sign of the polarized centre word. In addition, amplifiers become de-amplifiers if there are an odd number of negators within the context cluster. Similarly, de-amplifiers will become amplifiers if there are an even number of negators.
- Adversative conjunctions that appear before the polarized centre word up-weight the polarity of the context cluster, while adversative conjunctions that appear after the polarized centre word will downweight the polarity of the con-

text cluster.

For an amplification/de-amplification weight, z_1 (default value 0.8), and an adversative conjunction weight, z_2 (default value 0.85), the raw sentiment of a sentence is the sum of the weighted context clusters and is given by:

$$\sum c_{i,j} = \sum [(1 + \phi_a + \phi_d) \cdot |w_{i,j}^p| \cdot (-1)^{2+\phi_{neg}}]$$

where:

- $\phi_a = (\phi_{adv} > 1) + \sum(\phi_{neg} \cdot z_1 \cdot w_{i,j}^a)$ is the amplification weight
- $\phi_d = (\phi_{adv} < 1) + \max\{\sum z_1(-\phi_{neg} \cdot w_{i,j}^a + w_{i,j}^d), -1\}$ is the de-amplification weight
- $\phi_{neg} = (\sum w_{i,j}^{neg}) \bmod 2$ is the negation weight
- $|w_{i,j}^p|$ is the weight of the polarized centre word (obtained from the sentiment lexicon)
- $\phi_{adv} = 1 + z_2(n_{adv, before} - n_{adv, after})$ is the adversative conjunction weight, $n_{adv, before}$ is the number of adversative conjunctions before the polarized centre, $n_{adv, after}$ is the number of adversative conjunctions after the polarized centre

Finally, the raw sentiment scores of each sentence are scaled by dividing by the square root of the word count of the respective sentence.

4.3 Methods

Two sentiment lexicons were used to calculate tweet sentiments, both of which are available in the `lexicon`³¹ package:

1. Jockers-Rinker lexicon: a combination of Jockers' lexicon⁴² and Rinker's aug-

mented version of Hu and Liu’s lexicon^{42,43}. This lexicon contained a total of 11,710 entries, of which 287 were bigrams (a pair of terms). Entries had weights ranging from -2.0 to 1.0. The term `trump` was removed from this lexicon due to the fact that most, if not all, occurrences in this context would be in reference to Donald Trump rather than the verb.

2. NRC lexicon: a lexicon by National Research Council Canada’s Mohamamad and Turney⁴⁴. This lexicon contained a total of 5,468 entries, all of which were single terms. Entries had weights of -1 or +1. As the term `trump` did not appear in this lexicon, no additional modifications were required.

For the analysis presented in the next section, the sentiment of a tweet was computed as the sum of the raw sentiment scores. Raw sentiment scores were used rather than scaled sentiment scores since tweets with extreme sentiment scores were of interest and scaling would reduce the extremity of the sentiment scores, particularly for longer tweets and tweets that were less concise.

4.4 Results

Monthly average tweet sentiments were computed using the modified Jockers-Rinker lexicon (with the aforementioned removal of the term `trump`) and the NRC lexicon for the entire collection of tweets (Figure 4.1). Despite the difference in polarized term weightings and abundance of terms between the two lexicons, the overall monthly trends were quite similar. For both lexicons, the monthly average sentiments began very negatively in March. By May, the monthly average tweet sentiment had reached its least negative point (still below zero), followed immediately by a stark descent until October, and becoming slightly less negative in November than October. Interestingly, the average tweet sentiment for October surpassed the average tweet sentiment of March in negativity, when using the NRC lexicon.

Figure 4.1: Monthly average sentiments for tweets, using the Jockers-Rinker and NRC lexicons. Point sizes represent the proportion of tweets contributed by the given month, relative to other months.

Following from the trends of Figure 4.1, the five most negative tweets of March (Figure 4.2) and October (Figure 4.3) for both lexicons were extracted along with their corresponding cleaned texts. Collectively, the lowest scoring tweets of March discussed pandemic comparisons, criticisms of Trump and the American Republican party, the New York State COVID-19 case counts, and criticism of Indian political leaders. In October, the lowest scoring tweets collectively discussed criticisms of Trump and the American Republican party, and pandemic related frustrations. Interestingly, with the Jockers-Rinker lexicon, the lowest scoring tweet of October scores only one unit less than the lowest scoring tweet of March. Meanwhile, with the NRC lexicon, the lowest scoring tweet of October is almost twice as negative as the lowest scoring tweet of March.

4.5 Discussion

Upon examining the texts in Figures 4.2 and 4.3, it is apparent that a prominent issue with dictionary-based sentiment analyses on informal texts is the lack of consideration for excessive repetition. In particular, by repeating polarized terms, tweets are able to receive inflated sentiment scores while offering little semantic value to human readers. To improve the quality of the computed sentiment scores, one could implement a penalty for the repetition of polarized words occurring within a fixed window.

A caveat of dictionary-based sentiment analysis is that the choice of lexicon can be quite arbitrary and the term weightings of the chosen lexicon may not be appropriate for all contexts. In the setting of COVID-19, the term **positive** should have a positive weighting, whereas the bigram **test positive** should have a negative weighting. As the term **test** is unweighted in both Jockers-Rinker and NRC lexicons, a phrase where a user claims to have tested positive would be assigned a positive score.

From the plots of the monthly trends (Figure 4.1), it was seen that the average sentiments were all negative. However, without further investigation into the contents of each tweet, it cannot be concluded that the overall opinions of Twitter users *towards* the COVID-19 pandemic was negative. As was seen in Figures 4.2 and 4.3, few tweets managed to form an opinion on the COVID-19 pandemic. Instead, tweets mostly consisted of criticisms of political and societal actions in response to the pandemic.

Chapter 5

Topic modelling

5.1 Introduction

Topic models are statistical models used to learn the latent topics present in a collection of documents, also known as a *corpus*. Popular text data sources can include, but are not limited to, journal articles, news articles, social networking platforms, survey answers, and product reviews. In today’s world, copious amounts of text data are readily available in digitized formats – too much for even humans to process manually through reading. In the domain of information retrieval, topic modelling can help simplify the synthesis of large corpora. The training of a topic model on large text corpora can provide insight into the topical structures of the text, creating a structured framework towards understanding unstructured text.

In the setting of Twitter data, while hashtags have the potential to act as topic identifiers, the non-mandatory nature of hashtags makes the use of hashtags as topic identifiers infeasible. As discussed in the exploratory analysis, tweets that mentioned the term `trump` were not the only tweets about Trump. The appearance of the term `trump` in subfigures (b) through (f) of [Figure 3.6](#) also suggested that a topic such as

“Trump” would be very broad. As such, the use of single terms as topic identifiers is inadequate.

5.2 Latent Dirichlet Allocation (LDA)

To date, one of the most well known approaches to topic modelling is Latent Dirichlet Allocation (LDA), which proposes that documents cover a limited number of topics and topics tend to use a limited number of keywords⁴⁵. With the LDA approach, topic-term distributions and document-topic proportions are learned by analysing the document-term co-occurrences^{46,47}. Working with formal sources of text such as books, journal articles, and news articles, LDA has seen great success⁴⁵.

The Dirichlet distribution

The Dirichlet distribution is a multivariate continuous probability distribution parameterized by $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_K)$, with $\alpha_1, \dots, \alpha_K > 0$ and $K \geq 2$, and has probability density function:

$$f(x_1, \dots, x_K) = \frac{\Gamma(\sum_{i=1}^K \alpha_i)}{\prod_{i=1}^K \Gamma(\alpha_i)} \prod_{i=1}^K x_i^{\alpha_i-1},$$

$$x_1, \dots, x_K \geq 0, \quad \sum_{i=1}^K x_i = 1,$$

where $\Gamma(\cdot)$ is the gamma function. In the case where all elements of $\boldsymbol{\alpha}$ are equal, the distribution can be parameterized by the single common value α . The probability density function of the Dirichlet distribution can be simplified as:

$$f(x_1, \dots, x_K) = \frac{\Gamma(\alpha K)}{\Gamma(\alpha)^K} \prod_{i=1}^K x_i^{\alpha-1},$$

and is referred to as the symmetric Dirichlet distribution.

5.2.1 Model outline

The LDA model is constructed under the premise that each document is composed of a latent distribution of topics, and each topic is composed of a latent distribution of terms. Consider a corpus, $\mathbf{C}$, of D documents covering K topics over W unique terms in the vocabulary.

Notation:

- The collection of all terms, t , as they appear in the corpus is denoted $\mathbf{W} = \{t_i\}_{i=1}^{N_t}$, where N_t is the total number of terms in the corpus.
- The vocabulary of unique terms is denoted $V = \{w_j\}_{j=1}^W$.
- The number of terms in each document, $d \in [1, D]$, is denoted N_d .
- $z \in [1, K]$ is a topic indicator variable.
- The distribution of topics within the documents, $P(z | d)$, is represented by the $D \times K$ matrix,

$$\boldsymbol{\theta} = \{\theta_d\}_{d=1}^D,$$

whose individual entries may be denoted $\theta_{d,k} = P(z = k | d)$.

- The distribution of terms within topics, $P(w | z)$, is represented by the $K \times W$ matrix,

$$\boldsymbol{\Phi} = \{\phi_k\}_{k=1}^K,$$

whose individual entries may be denoted $\phi_{k,w} = P(w | z = k)$.

For data with a multinomial distribution, the Dirichlet distribution is a conjugate prior. For the sake of mathematical convenience, LDA relies on the use of symmetric, conjugate Dirichlet priors. Let α and β be the hyperparameters for two symmetric Dirichlet priors. α represents the topic uniformity within documents – a low value of

α coincides with the belief that documents cover few topics. β represents the term uniformity within topics – a low value of β coincides with the belief that topics use few terms. The generative process of LDA is as follows:

1. For each topic, $k \in [1, K]$:
 - (a) Draw the topic-term distribution, $\phi_k \sim \text{Dirichlet}(\beta)$
2. For each document, $d \in [1, D]$:
 - (a) Draw the topic distribution, $\theta_d \sim \text{Dirichlet}(\alpha)$
 - (b) Draw the document length, $N_d \sim \text{Poisson}(\xi)$
3. For each term in the d^{th} document, $w_{d,n}$, $n \in [1, N_d]$:
 - (a) Draw a topic assignment, $z_{d,n} \sim \text{Multinomial}(\theta_d)$
 - (b) Draw the term from the topic assignment, $w_{d,n} \sim \text{Multinomial}(\phi_{z_{d,n}})$

A graphical representation of the generative process of LDA can be found in [Figure 5.1](#).

Figure 5.1: Graphical model representation of LDA. Nodes in the graph represent random variables; shaded nodes are observed variables. Plates denote replication, with the number of replicates given in the bottom right corner of the plate.

The joint distribution of the latent and observed variables and hyperparameters for the d^{th} document can be written as:

$$P(\mathbf{w}_d, \mathbf{z}_d, \theta_d, \Phi \mid \alpha, \beta) = \prod_{n=1}^{N_d} P(w_{d,n} \mid \phi_{z_{d,n}}) P(z_{d,n} \mid \theta_d) P(\theta_d \mid \alpha) P(\Phi \mid \beta)$$

The likelihood of document $\mathbf{w}_d$ is obtained by integrating over θ_d and Φ , and summing over $\mathbf{z}_d$:

$$P(\mathbf{w}_d | \alpha, \beta) = \iint \prod_{n=1}^{N_d} \sum_{\mathbf{z}_d} P(w_{d,n} | \phi_{d,n}) P(z_{d,n} | \theta_d) P(\theta_d | \alpha) P(\Phi | \beta) d\theta_d d\Phi$$

Finally, the likelihood of the corpus $\mathbf{C}$ is computed as the product of the likelihoods of the individual documents:

$$P(\mathbf{C} | \alpha, \beta) = \prod_{d=1}^D P(\mathbf{w}_d | \alpha, \beta) \quad (5.1)$$

5.2.2 Parameter estimation

The estimation of LDA parameters through direct maximization of the likelihood function is infeasible as the problem is:

1. Ill-posed – a unique solution is not guaranteed.
2. Non-identifiable – the topic labels are unconstrained; while the label switching does not impact model fit, it results in a complex posterior that is rampant with multiple equivalent likelihood modes.
3. Intractable – there does not exist an efficient optimization algorithm when faced with multiple equivalent likelihood modes.

As such, approximate inference is done using collapsed Gibbs sampling. Gibbs sampling is a sampling algorithm used for statistical inference when the true joint distribution of a collection of parameters is unknown or difficult to sample from, but a conditional distribution is known or easier to sample from. Collapsed Gibbs sampling is a special case of Gibbs sampling where one or more variables have been marginalized out, often simplifying the sampling distribution even further.

As the complex posterior is rampant with multiple equivalent modes, the collapsed Gibbs sampler is used as an optimization tool rather than to obtain a posterior sample. By performing collapsed Gibbs sampling over the marginal distributions, the intractabilities in the posterior can be avoided. After collapsed Gibbs sampling, the document-topic probabilities and topic-term probabilities can be estimated by their maximum posterior value.

Due to the use of conjugate Dirichlet priors⁴⁸, $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ are marginalized over. For term t , only its topic z_i needs to be sampled according to the conditional distribution^{39,49}:

$$P(z_i = k | \mathbf{z}_{(i)}, \mathbf{W}) \propto \frac{\left(n_{(i),k}^{(t)} + \beta\right) \left(n_{(i),d}^{(k)} + \alpha\right)}{\left(\sum_{j=1}^W n_k^{(j)} - 1 + W\beta\right) \left(\sum_{v=1}^K n_d^{(v)} - 1 + K\alpha\right)}, \quad (5.2)$$

where:

- $\mathbf{z}_{(i)}$ is the topic assignments for all terms, except the current term, t
- $n_{(i),k}^{(t)}$ is the number of times the term t is assigned to topic k , excluding the current topic assignment
- $n_{(i),d}^{(k)}$ is the number of terms in document d assigned to topic k , excluding the current topic assignment
- $\sum_{j=1}^W n_k^{(j)} - 1$ is the total number of terms assigned to topic k , excluding the current topic assignment
- $\sum_{v=1}^K n_d^{(v)} - 1$ is the total number of terms in document d , excluding the current term, t

The collapsed Gibbs sampling algorithm for LDA ([Algorithm 5.1](#)) can be summarised

Algorithm 5.1 Collapsed Gibbs sampling algorithm for LDA.

Input: Number of topics K , Dirichlet hyperparameters α and β , collection of terms $\mathbf{W}$

Output: $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\Phi}}$

Randomly assign a topic to all terms $t \in \mathbf{W}$

for $iter = 1$ to N **do**

for each document, $d \in [1, D]$ **do**

for each term, t , in document d **do**

 Update topic assignment of t according to Equation 5.2

end for

end for

 Update $n_d^{(k)}, n_k^{(w)}$

end for

Compute estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ using Equation 5.3 and Equation 5.4, respectively

as follows⁴⁹. To begin, all terms in the corpus are randomly assigned a topic. For each iteration, 1 to N , the topic assignments of each term within each document are updated sequentially according to Equation 5.2. After N iterations, the number of times document d was assigned to topic k , $n_d^{(k)}$, and the number of times term w of the vocabulary was assigned to topic k , $n_k^{(w)}$ are counted. Finally, maximum conditional posterior estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ are computed using⁴⁹:

$$\hat{\theta}_{d,k} = \frac{n_d^{(k)} + \beta}{\sum_{v=1}^K n_d^{(v)} + K\alpha} \quad (5.3)$$

$$\hat{\phi}_{k,w} = \frac{n_k^{(w)} + \beta}{\sum_{j=1}^W n_k^{(j)} + W\beta} \quad (5.4)$$

5.2.3 Perplexity

Perplexity is a method of evaluating language model performance on held-out data and is equivalent to the geometric mean per-word profile likelihood³⁹:

$$\text{Perplexity}(\mathbf{C}_H | \hat{\boldsymbol{\theta}}_H, \hat{\boldsymbol{\Phi}}) = \exp \left\{ \frac{\log \left(P(\mathbf{C}_H | \hat{\boldsymbol{\theta}}_H, \hat{\boldsymbol{\Phi}}) \right)}{\sum_{d=1}^D \sum_{w=1}^W n^{(d,w)}} \right\},$$

with the profile likelihood in the numerator calculated using⁵⁰:

$$\log \left(P(\mathbf{C}_H | \hat{\boldsymbol{\theta}}_H, \hat{\boldsymbol{\Phi}}) \right) = \sum_{d=1}^D \sum_{w=1}^W n^{(d,w)} \log \left\{ \sum_{k=1}^K \hat{\theta}_{H,d,k} \hat{\phi}_{k,w} \right\},$$

where:

- $\mathbf{C}_H$ is the held-out corpus
- $\hat{\boldsymbol{\theta}}_H$ contains the document-topic proportions of the held-out data, estimated by repeating the collapsed Gibbs sampling procedure while keeping $\hat{\boldsymbol{\Phi}}$ fixed
- $n^{(d,w)}$ is the number of times the w^{th} term occurs in the d^{th} document

A low perplexity indicates that the model performs well as it is unsurprised by the term co-occurrences in the held-out data.

5.3 Biterm Topic Model (BTM)

With short texts such as tweets, if each tweet is treated as a single document, document-term co-occurrences can become extremely sparse and LDA can encounter difficulties learning topic-term relationships⁵¹. Some attempts to overcome this hurdle involved the creation of longer pseudo-documents through aggregation of users' tweets⁵² and aggregation of tweets sharing common terms⁵¹ prior to using LDA. These methods have experienced some improvement over LDA, but each subjective aggregation has its own disadvantages.

In 2013, the biterm topic model (BTM)⁵³ was introduced, which attempted to address the inadequacies of using LDA on short documents through the modelling of global term co-occurrences rather than at the document level. Today, BTM is one of the most well-known methods for topic modelling with short texts, with many variations building off of the core ideas of BTM such as FastBTM⁵⁴ and GraphBTM⁵⁵.

5.3.1 Model outline

To begin, biterms are extracted from the individual documents using a specified window size. The extracted biterms are unordered term pairs occurring within the document and the specified window size. For example, the single document containing terms

A B C B

and using a window size of 3 would result in two text windows, A B C and B C B, and produce biterms A B, B C, A C, B C, C B, and B B. As the term pairs are unordered, the biterm quantities can be summarised as: A B ($n = 1$), A C ($n = 1$), B C ($n = 3$), and B B ($n = 1$).

The BTM is constructed under the premise that terms co-occurring together frequently likely belong to the same topic. Therefore, the first step in learning the latent topic components is to model the generation of biterms. It is assumed that the topic is sampled from a mixture of topics over the entire corpus and each biterm is drawn from a specific topic independently, i.e. both terms are drawn from the same topic⁴⁶.

Given a corpus with D documents, suppose it contains N_B biterms, and K topics over W unique terms in the vocabulary.

Notation:

- The collection of biterms is denoted as $\mathbf{B} = \{b_i\}_{i=1}^{N_B}$
- The individual biterms are denoted as $b_i = (w_{i,1}, w_{i,2})$, where $w_{i,1}$ and $w_{i,2}$ are the terms of the biterm.
- $z \in [1, K]$ is a topic indicator variable.
- The distribution of topics within the corpus, $P(z)$, is represented by a K -

dimensional multinomial distribution,

$$\boldsymbol{\theta} = \{\theta_k\}_{k=1}^K, \quad \theta_k = P(z = k) \quad \text{and} \quad \sum_{k=1}^K \theta_k = 1$$

- The distribution of terms within topics, $P(w | z)$, is represented by the $K \times W$ matrix, Φ , with entries

$$\phi_{k,w} = P(w | z = k), \quad \text{where} \quad \sum_{w=1}^W \phi_{k,w} = 1$$

Like LDA, BTM relies on the use of symmetric, conjugate [Dirichlet](#) priors. Let α and β be the two hyperparameters for the Dirichlet priors. α represents the topic uniformity within the corpus, which in turn affects the topic uniformity within documents – a low value of α coincides with the belief that documents cover few topics. β represents the term uniformity within topics – a low value of β coincides with the belief that topics use few terms. The generative process of BTM is as follows:

1. Draw the global topic distribution, $\boldsymbol{\theta} \sim \text{Dirichlet}(\alpha)$
2. For each topic, $k \in [1, K]$:
 - (a) Draw the topic-term distribution, $\phi_k \sim \text{Dirichlet}(\beta)$
3. For each biterm, $b_i \in \mathbf{B}$:
 - (a) Draw a topic assignment, $z_i \sim \text{Multinomial}(\boldsymbol{\theta})$
 - (b) Draw two terms from the topic assignment, $w_{i,1}, w_{i,2} \sim \text{Multinomial}(\phi_{z_i})$

For simplicity, it is assumed that biterms are generated independently. A graphical representation of the generative process of BTM can be found in [Figure 5.2](#).

The probability of the biterm, $b_i = (w_{i,1}, w_{i,2})$, conditioned on the model param-

Figure 5.2: Graphical model representation of BTM. Nodes in the graph represent random variables; shaded nodes are observed variables. Plates denote replication, with the number of replicates given in the bottom right corner of the plate.

eters θ and Φ is given by:

$$\begin{aligned}
 P(b_i | \theta, \Phi) &= \sum_{k=1}^K P(w_{i,1}, w_{i,2}, z_i = k | \theta, \Phi) \\
 &= \sum_{k=1}^K P(z_i = k | \theta_k) P(w_{i,1} | z_i = k, \phi_{k,w_{i,1}}) P(w_{i,2} | z_i = k, \phi_{k,w_{i,2}}) \\
 &= \sum_{k=1}^K \theta_k \phi_{k,w_{i,1}} \phi_{k,w_{i,2}}
 \end{aligned}$$

Given the hyperparameters α and β , the probability of the biterm, b_i , is obtained by integrating over θ and Φ :

$$\begin{aligned}
 P(b_i | \alpha, \beta) &= \iint P(b_i | \theta, \Phi) P(\theta | \alpha) P(\Phi | \beta) d\theta d\Phi \\
 &= \iint \sum_{k=1}^K \theta_k \phi_{k,w_{i,1}} \phi_{k,w_{i,2}} P(\theta | \alpha) P(\Phi | \beta) d\theta d\Phi
 \end{aligned}$$

The likelihood of the entire corpus, $\mathbf{B}$, can be obtained by multiplying the probabilities of the individual biterms:

$$P(\mathbf{B} | \alpha, \beta) = \prod_{i=1}^{N_B} \iint \sum_{k=1}^K \theta_k \phi_{k,w_{i,1}} \phi_{k,w_{i,2}} P(\theta | \alpha) P(\Phi | \beta) d\theta d\Phi \quad (5.5)$$

The likelihood of the corpus in BTM (Equation 5.5) shares similarities with the likelihood of the corpus in LDA (Equation 5.1) except that BTM models biterm generation at the corpus level while LDA models term generation at the document

level.

5.3.2 Parameter estimation

As with LDA, the solving of the coupled parameters $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ through direct maximization of the likelihood function is ill-posed, non-identifiable, and intractable (Section 5.2.2). Therefore, approximate inference is carried out using collapsed Gibbs sampling.

Like LDA, the collapsed Gibbs sampler is used as an optimization tool rather than to obtain a posterior sample. Due to the use of conjugate Dirichlet priors^{46,48}, $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ are marginalized over. For biterm b_i , only its topic z_i needs to be sampled according to the conditional distribution^{46,56}:

$$P(z_i = k | \mathbf{z}_{(i)}, \mathbf{B}) \propto (n_{(i),k} + \alpha) \frac{(n_{(i),w_{i,1}|k} + \beta)(n_{(i),w_{i,2}|k} + \beta)}{(\sum_{j=1}^W n_{(i),j|k} + W\beta + 1)(\sum_{j=1}^W n_{(i),j|k} + W\beta)}, \quad (5.6)$$

where:

- $\mathbf{z}_{(i)}$ is the topic assignments for all biterms, excluding b_i
- $n_{(i),k}$ is the number of biterms assigned to topic k , excluding b_i
- $n_{(i),j|k}$ is the number of times term j is assigned to topic k , excluding b_i

Algorithm 5.2 Collapsed Gibbs sampling algorithm for BTM.

Input: Number of topics K , Dirichlet hyperparameters α and β , collection of biterms $\mathbf{B}$

Output: $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\Phi}}$

Randomly assign a topic to all biterms, $b_i = (w_{i,1}, w_{i,2}) \in \mathbf{B}$

for $iter = 1$ to N **do**

for each biterm, b_i **do**

 Update topic assignment of b_i according to Equation 5.6

 Update $n_k, n_{w_{i,1}|k}, n_{w_{i,2}|k}$

end for

end for

Compute estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ using Equation 5.7 and Equation 5.8, respectively

The collapsed Gibbs sampling algorithm for BTM ([Algorithm 5.2](#)) is summarised as follows⁴⁶. To begin, all biterms in the corpus are randomly assigned a topic. For each iteration, 1 to N , the topic assignments of each biterm are updated sequentially according to [Equation 5.6](#). After N iterations, the number of biterms in each topic, n_k , and the number of times term w was assigned to topic k , $n_{w|k}$, are counted. Finally, maximum conditional posterior estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ are computed using⁴⁶:

$$\hat{\theta}_k = \frac{n_k + \alpha}{N_B + K\alpha} \quad (5.7)$$

$$\hat{\phi}_{k,w} = \frac{n_{w|k} + \beta}{\sum_{j=1}^W n_{j|k} + W\beta} \quad (5.8)$$

5.3.3 Document-topic proportions

As the BTM models the generative process of biterms within the corpus, unlike LDA which models the generative process of terms within documents, the topic proportions of documents in BTM cannot be found directly. Instead, it is assumed that the topic proportions of a document equals the expectation of the topic proportions of biterms generated from the document^{46,53}:

$$P(z | d) = \sum_{b_i \in \mathbf{B}_d} P(z | b_i) P(b_i | d),$$

where $\mathbf{B}_d$ is the collection of biterms found in document d . The first term of the right-hand side, $P(z | b_i)$, is obtained using Bayes' formula with the estimated parameters of the model:

$$P(z = k | b_i) = \frac{P(z) P(w_{i,1} | z) P(w_{i,2} | z)}{\sum_z P(z) P(w_{i,1} | z) P(w_{i,2} | z)}$$

$$= \frac{\hat{\theta}_k \hat{\phi}_{k, w_{i,1}} \hat{\phi}_{k, w_{i,2}}}{\sum_{k=1}^K \hat{\theta}_k \hat{\phi}_{k, w_{i,1}} \hat{\phi}_{k, w_{i,2}}},$$

and the quantity $P(b_i | d)$ is estimated empirically as:

$$\widehat{P(b_i | d)} = \frac{n_d(b_i)}{\sum_{b \in d} n_d(b)},$$

where $n_d(b)$ is the number of times biterm b appears in document d .

5.3.4 Log-likelihood

Perplexities should not be used with BTM models, unlike LDA models, as LDA optimizes the likelihood of word co-occurrences within documents, while BTM optimizes the likelihood of biterm occurrences within the corpus⁴⁶. For BTM models, the log-likelihood of held-out data is used and can be calculated as³⁵:

$$\ell(b | \hat{\theta}, \hat{\Phi}) = \log \left\{ \sum_{b_i \in \mathbf{B}_H} \sum_{k=1}^K \hat{\theta}_k \hat{\phi}_{k, w_{i,1}} \hat{\phi}_{k, w_{i,2}} \right\}$$

where:

- $\mathbf{B}_H$ is the collection of held-out biterms
- Each biterm can be expressed as $b_i = (w_{i,1}, w_{i,2})$

A higher log-likelihood indicates that the model performs better as it is unsurprised by the biterm occurrences in the held-out data.

5.4 Comparison of LDA and BTM models

The performance of LDA and BTM models can be compared by comparing the quality of the learned topics. Topic models can often learn nonsense topics or topics that do not agree with human judgements, especially as the number of topics in a topic model

grows. The quality of the topics learned by LDA and BTM models can be assessed by calculating topic coherence scores using⁵⁷:

$$C(t; V^{(t)}) = \sum_{m=2}^M \sum_{l=1}^{m-1} \log \left(\frac{D(v_m^{(t)}, v_l^{(t)}) + 1}{D(v_l^{(t)})} \right),$$

where:

- t is the topic
- $V^{(t)} = (v_1^{(t)}, \dots, v_M^{(t)})$ is the list of the M most probable terms of topic t
- $D(v)$ is the number of documents where term v appears (document frequency of v)
- $D(v, v')$ is the number of documents where terms v and v' appear together (co-document frequency of v and v')

Informally, the coherence score posits that a topic is coherent if its M most probable terms co-occur in documents frequently. As such, a higher coherence score indicates that a topic is more coherent. While both LDA and BTM are constructed under the premise that terms co-occurring frequently likely belong to the same topic, as LDA encounters difficulties in learning topic-term relationships in shorter documents, it is unsurprising that BTM often achieves higher coherence scores than LDA^{53,58}.

5.5 Methods

5.5.1 LDA

Restarting from the cleaned data obtained at the end of the data preparation step (Chapter 2), further processing was performed prior to model fitting. Stopwords were removed according to a custom stopwords list, in order to decrease the number

of potential terms in the vocabulary. This custom stopwords list was derived from the `snowball` lexicon of stopwords found within the `tidytext`²² package, with the following modifications:

- Negation words were removed from the stopwords list, e.g., `not`, `can't`, `don't`, `against`, etc.
- Remaining contraction stopwords that do not become words when the apostrophe is missing were added to the stopwords list, e.g., `heres`, `theyll`, `hows`, etc. This was done to account for the fact that punctuation is not mandatory on Twitter and to reduce vocabulary size. As an example, words such as `ill`, `hell`, `well`, originating from `i'll`, `he'll`, and `we'll`, were not added to the stopwords list.

The remaining terms were then stemmed using the `SnowballC`³³ package, according to the Porter stemming algorithm, once again in an effort to reduce the vocabulary size. For example, terms such as `want`, `wants`, and `wanted`, would all become `want`.

The hyperparameters that required tuning were K (number of topics), α (document-topic uniformity), and β (topic-term uniformity). Due to the size of the data, tuning of hyperparameters using cross-validation was not feasible. Instead, an 80/10/10 training/validation/testing split was made. In order to evaluate the trained models on the validation and testing sets, it was imperative that the validation and testing sets did not contain terms that were not seen in the training phase. Therefore, terms across the three splits that appeared in five or fewer tweets were removed.

The **R** package `topicmodels`³⁹ was used to fit the LDA models via collapsed Gibbs sampling. The tuning of the hyperparameters was as follows:

1. Fix K while varying α and β . For all combinations of α and β :
 - (a) Train a LDA model using training data.

- (b) Evaluate model on validation data to obtain a perplexity.
 - (c) Collect value of α and β that resulted in the lowest perplexity value.
2. Varying values of K , while fixing α and β to the values obtained from step 1:
 - (a) Train a LDA model using training data.
 - (b) Evaluate model on validation data to obtain a perplexity.
 - (c) Collect value of K that resulted in the lowest perplexity value.
 3. Evaluate final model performance by obtaining a perplexity using testing data.

Out of consideration for time and the number of models that were required to be trained, 50 iterations of collapsed Gibbs sampling were performed for each model.

5.5.2 BTM

The same processed text data and its corresponding splits used to fit the LDA models were reused for fitting the BTM models. The **R** package `BTM`³⁵ (a wrapper for the original **C++** implementation of the BTM by Yan et al.⁵³) was used to perform the biterm topic modelling. The biterm window size was kept at the default value of 15 as it was expected that most tweets would contain 15 or fewer terms following the removal of custom stopwords and rare terms in the data processing step. The hyperparameters that required tuning were K (number of topics), α (corpus-topic uniformity), and β (topic-term uniformity). Due to the size of the data, the tuning of hyperparameters with cross-validation was not feasible and a tuning process similar to that of LDA was taken:

1. Fix K while varying α and β . For all combinations of α and β :
 - (a) Train a BTM model using training data.

- (b) Evaluate model on validation data to obtain a log-likelihood.
 - (c) Collect value of α and β that resulted in the highest log-likelihood value.
2. Varying values of K , while fixing α and β to the values obtained from step 1:
- (a) Train a BTM model using training data.
 - (b) Evaluate model on validation data to obtain a log-likelihood.
 - (c) Collect value of K that resulted in the highest log-likelihood value.
3. Evaluate final model performance using testing data.

Out of consideration for time and the number of models that were required to be trained, 50 iterations of collapsed Gibbs sampling were performed for each model.

5.6 Results

5.6.1 LDA

Model tuning

In the first step of the tuning process, the following values were used:

$$K = 10, \quad \alpha = \{1/K, 50/K, 100/K\}, \quad \beta = \{0.001, 0.01, 0.1\}$$

Nine models were trained on the training data and evaluated on the validation data to obtain perplexities ([Figure 5.3](#)). The model with the lowest perplexity with $K = 10$ fixed, was the model using $\alpha = 100/K$ and $\beta = 0.1$.

In the second step of the tuning process, the following values were used:

$$K = \{20, 50, 100, 200, 300, 400, 500\}, \quad \alpha = 100/K, \quad \beta = 0.1$$

Figure 5.3: Perplexities for the nine models trained in the first tuning step, evaluated on the validation set. K , the number of topics, was fixed while varying values of α and β . The hyperparameters corresponding to the model with the lowest perplexity were $\alpha = 100/K$ and $\beta = 0.1$.

Seven models were trained on the training data and evaluated on the validation data to obtain perplexities (Figure 5.4). The perplexities continued to decrease with an increase in number of topics (K). The perplexity for the $K = 500$ model was unable to be calculated due to insufficient memory. It is possible that the perplexities would have continued to decrease for values of K greater than 500. As the perplexity for the $K = 500$ model evaluated on the validation set was unable to be calculated due to memory constraints, the perplexity for the same model evaluated on the testing set was also unable to be calculated. Due to the aforementioned memory constraints, it was decided that no additional models would be trained. As such, the $K = 500$ model was taken to be the optimal model. Henceforth, “model” will refer to the $K = 500$ model unless otherwise specified.

Figure 5.4: Perplexities for six of the seven models trained in the second tuning step, plus the best model from the first tuning step, evaluated on the validation set. α and β were fixed, while varying over values of K . Perplexities involving the $K = 500$ model were unable to be calculated due to memory constraints.

Document-topic allocations

As each document is assumed to contain a mixture of topics, document-topic proportions were treated as topic allocations and summed for each topic by month. Monthly topic allocations were then scaled by dividing by the number of tweets available for each month, so as to not artificially inflate topic allocations due to having more tweets in a given month.

The five topics with the highest scaled allocations overall were highlighted in Figure 5.5. Using Figure 5.7 as a guide, the five topics shown in Figure 5.5 were labelled.

For the months of March and April, all five of the highlighted topics had extremely low scaled allocation values. Throughout May and June, the topic of police brutality

Figure 5.5: Monthly document-topic allocations scaled by the number of tweets available per month. The five topics highlighted attained the highest scaled allocation values overall.

had the highest scaled allocation, corresponding to the murder of George Floyd and the subsequent protests on police brutality towards blacks. The postponement of Indian NEET and JEE entrance exams attained the highest scaled allocation in the month of August, followed immediately by a sharp decline in subsequent months. In September, the topic with the highest scaled allocation concerned the swarming of the Korean boy-band, Enhypen, by fans after the leakage of their private flight schedule. Finally, topics discussing the USA presidential election and voting methods saw slow increases in scaled allocation from month to month, but eventually attained the highest scaled allocations for the months of October and November, coinciding with the election which took place in November 2020.

The scaled topic allocations were ranked for each month and the top three topics

Figure 5.6: Monthly top three topics by scaled allocations.

for each month were displayed in Figure 5.6. Once again, using Figure 5.7 as a guide, the topics that appeared in Figure 5.6 were labelled. Interestingly, there were two topics in the months of June and July that gave advice to the general public. While the top terms for both topics were quite similar, one topic placed an emphasis on hand-washing, while the other placed an emphasis on wearing masks.

Figure 5.7: Top 20 most probable terms for topics appearing in Figure 5.5 and Figure 5.6.

5.6.2 BTM

Model tuning

In the first step of the tuning process, the following values were used:

$$K = 10, \quad \alpha = \{1/K, 50/K, 100/K\}, \quad \beta = \{0.001, 0.01, 0.1\}$$

Figure 5.8: Log-likelihoods for the nine models trained in the first tuning step, evaluated on the validation set. K , the number of topics, was fixed at 10 while varying values of α and β . The hyperparameters corresponding to the model with the highest log-likelihood were $\alpha = 50/K$ and $\beta = 0.1$.

Nine models were trained on the training data and evaluated on the validation data to obtain log-likelihoods (Figure 5.8). The model with the highest log-likelihood with $K = 10$ fixed, was the model using $\alpha = 50/K$ and $\beta = 0.1$.

In the second step of the tuning process, the following values were used:

$$K = \{20, 50, 100, 200, 300, 400, 500\}, \quad \alpha = 50/K, \quad \beta = 0.1$$

Figure 5.9: Log-likelihoods for the seven models trained in the second tuning step, plus the best model from the first tuning step, evaluated on the validation set (black) and the testing set (green). α and β were fixed, while varying over values of K .

Seven models were trained on the training data and evaluated on the validation data to obtain log-likelihoods (Figure 5.9). The log-likelihoods continued to increase with an increase in number of topics (K). It is possible that the log-likelihood would have continued to increase for values of K greater than 500. However, due to the fact that the training of the model with $K = 500$ took slightly more than two weeks to complete, it was decided that no additional models would be trained. As such, the $K = 500$ model was taken to be the optimal model and evaluated on the testing data to obtain a final log-likelihood. Henceforth, “model” will refer to the $K = 500$ model unless otherwise specified.

Document-topic allocations

As each document is assumed to contain a mixture of topics, document-topic proportions were treated as topic allocations and summed for each topic by month. Monthly topic allocations were then scaled by dividing by the number of tweets available for each month, so as to not artificially inflate topic allocations due to having more tweets in a given month.

Figure 5.10: Monthly document-topic allocations scaled by the number of tweets available per month. The five topics highlighted obtained the highest scaled allocation values overall.

The five topics with the highest scaled allocations overall were highlighted in Figure 5.10. Using Figure 5.12 and Figure 5.13 as guides, the five topics shown in Figure 5.10 were labelled.

In March, the highest scaled allocation was attributed to the topic discussing the American presidential election. From its corresponding biterm cluster (Figure 5.12),

it was suggested that discussions were motivated by Trump’s handling of the COVID-19 pandemic, the responsibilities (word stem: **respons**) of a capable president, and support for the opposing candidate, Joe Biden. As the months passed, the topic of the American presidential election slowly saw an increase in its scaled allocation, eventually reaching the highest monthly scaled allocation for three straight months (September to November). Unsurprisingly, another election-related topic which discussed voting methods saw a slow growth in its monthly scaled allocation until the month of the election (November) where it attained the second highest scaled allocation.

The topics of general pandemic advice and shopping saw large growths in scaled allocation from March to April, and maintained the highest scaled allocations for the months of April to July, eventually being overtaken by the election topics. Discussions on the postponement of the Indian NEET and JEE entrance exams saw a sharp increase in scaled allocation in the month of August, attaining the highest scaled allocation for the month. Immediately after, this topic saw a sharp decrease in scaled allocation, eventually becoming indistinguishable among the other unpopular topics.

The scaled topic allocations were ranked for each month and the top three topics for each month were displayed in [Figure 5.11](#). Using [Figure 5.12](#) and [Figure 5.13](#) once again as guides, the topics that appeared in [Figure 5.11](#) were labelled.

From the biterm cluster of the topic discussing the lack of masks at republican events ([Figure 5.12](#)), the biterm with the highest number of occurrences was **no mask**. Due to the presence of this biterm, one would have likely described this topic as tweets in support of not wearing masks. However, upon further analysis into the tweets with the highest allocations for this topic, it was found that this topic arose from tweeters calling out the lack of masks at Republican events. In June, this topic was the third most discussed topic and coincided with Trump’s rally in Tulsa, Oklahoma.

Figure 5.11: Monthly top three topics by scaled allocations.

For visual clarity, the `textplot_bitermclusters` function from the `textplot`³⁶ package requires terms and biterms to appear at most once when plotting multiple topics in a single call in order to prevent overlapping clusters. A term that is common to more than one of the topics to be plotted is reserved for the single topic that has the highest topic-term probability for the term. Edges joining terms to form biterms are only plotted within a topic cluster if both component terms appear in the cluster and if the biterm had the highest number of assignments towards the particular topic, in comparison to the other topics to be plotted.

In Figure 5.12, the topics of “social distancing enforcement and violations”, “shopping”, “no masks at republican events” and “general pandemic advice” all had `social` and `distanc` appear in their top 20 terms (Figure 5.13). Out of these four topics, `social` and `distanc` both had the highest topic-term probabilities in the topic of “social distancing enforcement and violations” and therefore appeared as nodes in

this cluster. The biterm `social distanc` had the most assignments towards the topic “general pandemic advice”. As a result, the nodes `social` and `distanc` in the “social distancing enforcement and violations” were not joined by an edge. For the sake of convenience and visual aesthetics, the clusters in [Figure 5.12](#) were plotted in a single call rather than plotting the individual component plots separately and stitching them back together.

Biterm topic model: $K = 500$

Figure 5.12: Biterm clusters for topics appearing in Figure 5.10 and Figure 5.11. Each cluster is an undirected graph as biterms are unordered pairs of terms. Increased node (term) size corresponds to higher topic-term probability. Increased thickness and darkness of edges (links) corresponds to higher co-occurrences within the topic. As biterms were computed using a window size of 15, adjacent nodes may not have necessarily appeared adjacently in the original text.

Figure 5.13: Top 20 most probable terms for topics appearing in [Figure 5.10](#) and [Figure 5.11](#).

5.6.3 Topic quality

Figure 5.14: Mean coherence scores and 95% *t*-intervals for the 500 topics of the BTM and LDA models using the top $M = \{5, 10, 15, 20\}$ terms of each topic, evaluated on the testing data. A higher coherence score means that a topic is more coherent. Using a two-sample *t*-test, the mean coherence of topics learned by BTM was found to be significantly ($p < 0.001$) greater than the mean coherence of topics learned by LDA for the each of the four values of M considered here.

For each topic within each model, the top $M = \{5, 10, 15, 20\}$ terms were used to calculate the coherence scores. For each value of M and model, the coherence scores of all topics evaluated on the testing data were averaged with a 95% *t*-interval constructed around the centre (Figure 5.14). Using a small number for M , the topic qualities of BTM and LDA were comparable. However, as the number of top words increased, the distinction of the mean coherence between the models became more apparent. It was also seen that the individual topic coherence scores of BTM exhibited much less variability compared to LDA. By two-sample *t*-test, it was found that the mean topic coherence of BTM was significantly ($p < 0.001$) greater than the mean

topic coherence of LDA for each of the four values of M used considered here. This was consistent with the results presented in Yan et al.⁵³.

5.7 Discussion

It was seen that three of the five overall top topics from March to November were shared by LDA and BTM: the USA presidential election, voting methods in the USA presidential election, and the postponement of the Indian entrance exams. The differences in topic allocations between the two models became apparent when looking at the top three topics for each month. In LDA, many of the top monthly topics were not the same as the overall top topics, whereas for BTM, many of the top monthly topics were the same as the overall top topics. Nonetheless, the use of both LDA and BTM has provided insightful information into the topics discussed in these COVID-19 tweets.

The application of topic modelling on Twitter data comes with some caveats. When using Twitter data, it is assumed that the collected data has semantic value. Unfortunately, regardless of the number of steps taken to filter out tweets of low quality, it is likely impossible that all such cases will be found, especially for data of this size. Unknowingly having a large amount of low quality tweets will likely have a negative impact on the learned topics.

As the most important part of topic modelling is to allow for humans to interpret the resulting topics, when topics arise due to short, vague text data, interpretation may not always be possible. While Twitter users can collectively discuss specific topics which are successfully learned by the topic models, a lack of context combined with the short nature of representative tweets can often add to the difficulty of interpreting a topic, especially to outsiders.

To date, LDA has few visualizations that aid users in interpreting topics. As a result of modelling term co-occurrences at the corpus level, biterm topic clusters such as those shown in [Figure 5.12](#) prove to be indispensable visual aids for interpreting the learned topics. In conclusion, as the topics learned by BTM were, on average, more coherent than those learned by LDA ([Figure 5.14](#)), and easier to interpret with the aid of the `textplot`³⁶ package, it is clear that BTM remains superior to LDA when working with short-text data.

Chapter 6

Conclusions

In this thesis, many numerical and graphical methods were presented towards analysing informal, unstructured short-text data pertaining to the COVID-19 pandemic. This included analysing term frequencies and term co-frequencies, sentiment analysis, and the training of topic models in order to better understand the topics of discussions throughout the global pandemic. In regards to topic modelling, it was concluded that the topics learned by BTM were, on average, more coherent than the topics learned by LDA. In spite of these promising results, there are a few naturally arising shortcomings to consider in regards to topic models such as LDA and BTM.

First, held-out data cannot contain terms that were not seen in the training step. In the setting of Twitter data, this often involves removing terms that appeared in too few documents, which in many cases are unique hashtags or proper nouns. However, it is left up to the user to decide how many is “too few” and the resulting cut-off point can be somewhat arbitrary, varying between data sets. The possibility of out-of-vocabulary terms also means that previously trained models may not be re-usable for new text data of a similar theme without extensive pre-processing of the new text.

Second, the number of topics must be chosen before fitting LDA or BTM models.

While the true number of topics remains unknown, using too few topics will result in unhelpful and difficult to interpret topic clusters. On the other hand, using too many topics can result in computational difficulties due to a lack of computational resources. In addition, using too many topics defeats the purpose of topic modelling as the number of topics to interpret no longer becomes manageable.

Finally, there is a lack of justification for the use of Dirichlet priors other than for mathematical convenience.

Future work

In 2018, a novel approach to topic modelling was introduced by Gerlach et al.⁵⁹ which incorporated a community detection technique, known as stochastic block modelling, into the realm of topic modelling. In this study, it was concluded that their model, the hierarchical stochastic block model (hSBM), was able to outperform LDA on the basis of minimum description length⁶⁰ on both long and short-text data. In addition, the need to choose the number of topics was eliminated as it was done automatically, and the use of the non-justified Dirichlet priors was eliminated through the formulation of new priors. As the results of hSBM appear extremely promising, it is of interest to compare the performance of the hSBM with BTM for short-text data.

Bibliography

- [1] World Health Organization. Coronavirus disease (covid-19). URL <https://www.who.int/news-room/q-a-detail/coronavirus-disease-covid-19>.
- [2] Johns Hopkins University & Medicine. Johns hopkins coronavirus resource center. *Johns Hopkins Coronavirus Resource Center*. URL <https://coronavirus.jhu.edu/>.
- [3] Rob McLean, Laura He, and Anneken Tappe. Dow plunges 1,000 points, posting its worst day in two years as coronavirus fears spike. *CNN*. URL <https://www.cnn.com/2020/02/23/business/stock-futures-coronavirus/index.html>.
- [4] Coronavirus scare: Complete list of airlines suspending flights. *India Today*. URL <https://www.indiatoday.in/lifestyle/travel/story/coronavirus-scare-complete-list-of-airlines-suspending-flights-1650574-2020-02-27>.
- [5] James Brownsell. Ilo: Half of all workers risk losing jobs due to virus. URL <https://www.aljazeera.com/economy/2020/4/29/half-the-worlds-workers-face-losing-their-jobs-says-ilo>.
- [6] Jessica Guynn. Coronavirus fears empty store shelves of toilet paper, bottled water, masks as shoppers stock up. *USA TODAY*. URL <https://www.usatoday.com/story/money/2020/02/28/coronavirus-2020->

[preparation-more-supply-shortages-expected/4903322002/](#).

- [7] Nylah Burton. The coronavirus exposes the history of racism and “cleanliness”. *Vox*, Feb 2020. URL <https://www.vox.com/2020/2/7/21126758/coronavirus-xenophobia-racism-china-asians>.
- [8] Fears of coronavirus trigger anti-china sentiment worldwide. *koreatimes*, Feb 2020. URL http://www.koreatimes.co.kr/www/world/2021/01/683_282767.html.
- [9] Claire C. Miller. Could the pandemic wind up fixing what’s broken about work in america? *The New York Times*, Apr 2020. ISSN 0362-4331. URL <https://www.nytimes.com/2020/04/10/upshot/coronavirus-future-work-america.html>.
- [10] Ian Swanson. Five ways the coronavirus could change american politics. *The Hill*, May 2020. URL <https://thehill.com/homenews/campaign/495761-five-ways-the-coronavirus-could-change-american-politics>.
- [11] Brett Molina. Twitter overcounted active users since 2014, shares surge on profit hopes. *USA TODAY*. URL <https://www.usatoday.com/story/tech/news/2017/10/26/twitter-overcounted-active-users-since-2014-shares-surge/801968001/>.
- [12] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL <https://www.R-project.org/>.
- [13] Michael W. Kearney. rtweet: Collecting and analyzing twitter data. *Journal of Open Source Software*, 4(42):1829, 2019. doi: 10.21105/joss.01829. URL <https://joss.theoj.org/papers/10.21105/joss.01829>. R package version 0.7.0.

- [14] Travers Ching. *qs: Quick Serialization of R Objects*, 2020. URL <https://CRAN.R-project.org/package=qs>. R package version 0.23.3.
- [15] Hadley Wickham, Romain François, Lionel Henry, and Kirill Müller. *dplyr: A Grammar of Data Manipulation*, 2020. URL <https://CRAN.R-project.org/package=dplyr>. R package version 1.0.2.
- [16] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL <https://ggplot2.tidyverse.org>.
- [17] Lionel Henry and Hadley Wickham. *purrr: Functional Programming Tools*, 2020. URL <https://CRAN.R-project.org/package=purrr>. R package version 0.3.4.
- [18] Henrik Bengtsson. A unifying framework for parallel and distributed processing in r using futures, aug 2020. URL <https://arxiv.org/abs/2008.00553>.
- [19] Davis Vaughan and Matt Dancho. *furrr: Apply Mapping Functions in Parallel using Futures*, 2020. URL <https://CRAN.R-project.org/package=furrr>. R package version 0.2.1.
- [20] Kirill Müller and Hadley Wickham. *tibble: Simple Data Frames*, 2020. URL <https://CRAN.R-project.org/package=tibble>. R package version 3.0.4.
- [21] Hadley Wickham. *tidyr: Tidy Messy Data*, 2020. URL <https://CRAN.R-project.org/package=tidyr>. R package version 1.1.2.
- [22] Julia Silge and David Robinson. tidytext: Text mining and analysis using tidy data principles in r. *JOSS*, 1(3), 2016. doi: 10.21105/joss.00037. URL <http://dx.doi.org/10.21105/joss.00037>.
- [23] Hadley Wickham. *stringr: Simple, Consistent Wrappers for Common String Operations*, 2019. URL <https://CRAN.R-project.org/package=stringr>. R

package version 1.4.0.

- [24] Jeroen Ooms. *cld2: Google's Compact Language Detector 2*, 2018. URL <https://CRAN.R-project.org/package=cld2>. R package version 1.2.
- [25] Jeroen Ooms. *cld3: Google's Compact Language Detector 3*, 2020. URL <https://CRAN.R-project.org/package=cld3>. R package version 1.3.
- [26] Garrett Grolemund and Hadley Wickham. Dates and times made easy with lubridate. *Journal of Statistical Software*, 40(3):1–25, 2011. URL <https://www.jstatsoft.org/v40/i03/>.
- [27] Kamil Slowikowski. *ggrepel: Automatically Position Non-Overlapping Text Labels with 'ggplot2'*, 2020. URL <https://CRAN.R-project.org/package=ggrepel>. R package version 0.8.2.
- [28] Stefan Milton Bache and Hadley Wickham. *magrittr: A Forward-Pipe Operator for R*, 2020. URL <https://CRAN.R-project.org/package=magrittr>. R package version 2.0.1.
- [29] Ian Fellows. *wordcloud: Word Clouds*, 2018. URL <https://CRAN.R-project.org/package=wordcloud>. R package version 2.6.
- [30] Erich Neuwirth. *RColorBrewer: ColorBrewer Palettes*, 2014. URL <https://CRAN.R-project.org/package=RColorBrewer>. R package version 1.1-2.
- [31] Tyler W. Rinker. *lexicon: Lexicon Data*. Buffalo, New York, 2018. URL <http://github.com/trinker/lexicon>. version 1.2.1.
- [32] Tyler W. Rinker. *sentimentr: Calculate Text Polarity Sentiment*. Buffalo, New York, 2019. URL <http://github.com/trinker/sentimentr>. version 2.7.1.
- [33] Milan Bouchet-Valat. *SnowballC: Snowball Stemmers Based on the C 'libstemmer' UTF-8 Library*, 2020. URL <https://CRAN.R-project.org/package=>

- [SnowballC](#). R package version 0.7.0.
- [34] Max Kuhn, Fanny Chow, and Hadley Wickham. *rsample: General Resampling Infrastructure*, 2020. URL <https://CRAN.R-project.org/package=rsample>. R package version 0.0.8.
- [35] Jan Wijffels. *BTM: Biterm Topic Models for Short Text*, 2020. URL <https://CRAN.R-project.org/package=BTM>. R package version 0.3.4.
- [36] Jan Wijffels. *textplot: Text Plots*, 2020. URL <https://CRAN.R-project.org/package=textplot>. R package version 0.1.4.
- [37] Thomas Lin Pedersen. *ggraph: An Implementation of Grammar of Graphics for Graphs and Networks*, 2020. URL <https://CRAN.R-project.org/package=ggraph>. R package version 2.0.2.
- [38] Joël Gombin, Ramnath Vaidyanathan, and Vladimir Agafonkin. *concaveman: A Very Fast 2D Concave Hull Algorithm*, 2020. URL <https://CRAN.R-project.org/package=concaveman>. R package version 1.1.0.
- [39] Bettina Grün and Kurt Hornik. topicmodels: An R package for fitting topic models. *Journal of Statistical Software*, 40(13):1–30, 2011. doi: 10.18637/jss.v040.i13.
- [40] Hiroaki Yutani. *gghighlight: Highlight Lines and Points in 'ggplot2'*, 2020. URL <https://CRAN.R-project.org/package=gghighlight>. R package version 0.3.1.
- [41] Enhypen reportedly left hurt, scared, & crying after being mobbed by “fans” on first airport trip 2020. *Koreaboo*, Sep 2020. URL <https://www.koreaboo.com/news/enhypen-reportedly-left-hurt-scared-crying-after-mobbed-fans-first-airport-trip/>.
- [42] Matthew L. Jockers. *Syuzhet: Extract Sentiment and Plot Arcs from Text*, 2015. URL <https://github.com/mjockers/syuzhet>.

- [43] Mingqing Hu and Bing Liu. *Mining and summarizing customer reviews*. Seattle, Washington, 2004.
- [44] Saif M. Mohammad and Patrick D. Turney. *Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon*, 2010.
- [45] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4):993 – 1022, 2003.
- [46] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. BTM: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26: 2928 – 2941, 2014.
- [47] Xuerui Wang and Andrew McCallum. Topics over time: A non-markov continuous-time model of topical trends. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 424 — 433, 2006. doi: 10.1145/1150402.1150450.
- [48] Jun S. Liu. The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 89(427):958 — 966, Sep 1994. ISSN 0162-1459, 1537-274X. doi: 10.1080/01621459.1994.10476829.
- [49] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. *Proceedings of the 17th international conference on World Wide Web*, page 91–100, Apr 2008. doi: 10.1145/1367497.1367510.
- [50] David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, pages 1801 – 1828, 2009.

- [51] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in twitter. pages 80 — 88, 2010. doi: 10.1145/1964858.1964870.
- [52] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twiterrank: Finding topic-sensitive influential twitterers. pages 261 — 270, 2010. doi: 10.1145/1718487.1718520.
- [53] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. pages 1445 — 1456, May 2013. doi: 10.1145/2488388.2488514.
- [54] Xingwei He, Hua Xu, Jia Li, Liu He, and Linlin Yu. FastBTM: Reducing the sampling time for biterm topic model. *Knowledge-Based Systems*, 132:11 – 20, 2017. doi: 10.1016/j.knosys.2017.06.005.
- [55] Qile Zhu, Zheng Feng, and Xiaolin Li. Graphbtm: Graph enhanced autoencoded variational inference for biterm topic model. pages 4663 — 4672, 2018. URL <https://www.aclweb.org/anthology/D18-1495.pdf>.
- [56] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. Supplemental material of BTM: Topic modeling over short texts, 2014. URL <http://xiaohuiyan.github.io/paper/BTM-TKDE-supplemental.pdf>.
- [57] David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. pages 262 – 272, Jul 2011. URL <https://www.aclweb.org/anthology/D11-1024>.
- [58] Elias Jónsson and Jake Stolee. An evaluation of topic modelling techniques for twitter.
- [59] Martin Gerlach, Tiago P Peixoto, and Eduardo G Altmann. A network approach to topic models. *Science advances*, 4(7):eaq1360, 2018.
- [60] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465

– 471, 1978.