A comprehensive topic-model based hybrid sentiment analysis system

by

Yibing Yang

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Information Technology

in

Digital Media

Carleton University
Ottawa, Ontario

Abstract

Nowadays, Twitter sentiment analysis is drawing a lot of attention due to its potential to drive decision making in a variety of domains. However, the trend that publicly available training datasets are becoming less available, the difficulty in determining topic numbers for topic model-based approach, and a lack of data level discussion about how to utilize the proposed models day-to-day to drive applications are still the remained concerns. To solve these problems, we firstly offer a new method to collect and build Twitter training dataset based on noisy labels; In addition, we proposed a topic-model based hybrid sentiment classification model by using our self-collected tweets, which utilizes three different topic models and coherence score to choose the best topic model in an automated way; Last but not least, a use case is illustrated to show how to apply our pipeline in a daily basis to solve real business problems.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Omair Shafiq, for the continuous support of my two years master study, for his patience, motivation, and immense knowledge. During the two years, we have had a lot of meetings and email correspondence, and Prof. Shafiq is always being responding, helpful, and guide me based on his immense knowledge and experience in this domain. Every time after our conversation, I feel very inspired and know what to do next with passion and confidence. I'm so glad to have the chance to work with Prof. Shafiq on some exciting projects because without his precious support. It would not be possible for me to conduct this research.

My sincere thanks also go to Dr. Arya, Dr. Girouard, Dr. Joslin, Dr. Dehne, Dr. Hutchinson, Dr. Cheung, Dr. Baysal from whom I received directly or indirectly help and support. I really appreciate your guidance and knowledge, which makes me get out of my comfort zone and have the chance to learn art-of-the-state technologies.

I would like to extend my thanks to the entire administration and faculty of the School of Information Technology, who is always there listening to students and provides facilities and opportunities to us.

Table of Contents

List of Tables

List of Illustrations

Chapter 1: Background and Introduction

1.1 Background

Since the Web 2.0 era replaced Web 1.0 era several years ago, it brings a significant evolution to the ways people interact with the Internet. Instead of a quite static, read-only information portal (Web 1.0 era), we are embracing a more dynamic, read-write, interactable Internet environment. The fundamental changes on designs and services provided by current webpages and applications encourage user-generated contents than ever before and allow people to have a variety of channels to express their experiences, opinions, and sentiments freely.

In such an era that social media and we media thrive, not only the users but also the researchers can benefit from it. Knowing other people's sentiments is always a need that can facilitate decision making. While only starting from around ten years ago, people's sentiments could be recorded and stored in a large scale which makes it possible to be analyzed and understood further. Naturally, sentiment analysis starts to draw people's attention than ever before since it is the key method that can help people uncover the insights behind the text data.

It is clear that the applications built on top of sentiment analysis could benefit a variety of domains. From the perspective of commercial, for example, the buyers could rapidly take advantage of reviews from previous customers to measure the strength and weakness of the item they are interested; While for the merchants who are selling products, they could get a better understanding of what feature of the product is loved by their customers and which parts should be improved. The political domain could also benefit from sentiment analysis for requests like president election prediction, support rate

investigation for different partiers, public event analysis, and so on. Another domain that could be driven by sentiment analysis is public security. Cyber violence control, terrorism activities monitoring, suicide tendency detections are all potential use cases in this domain, since if we can capture signals in advance, we can take actions in a proactive way.

In our study, we want to focus on sentiment analysis in the commercial perspective. So far, a huge amount of opinionated data could be found online in terms of e-commerce platform and social networks. Due to Amazon customer review dataset, with an 18 years window (May 1996 - July 2014) around 142.8 million reviews are collected from a list of categories[1]; Another e-commerce platform eBay has 180 million active buyers with 1.2 billion live listings[2]; Twitter, a popular microblogging and social networking platform, has an average of 126 million daily active users and 321 million monthly active users for quarter 4, 2018[3] with over 500 million tweets being sent each day[4]. 93.3% of people who follow small and medium-sized businesses on Twitter plan to purchase from the brands they follow, and 68.7% have already purchased from a small and medium-sized business because of something they saw on the network[5]. Overall, a massive amount of customer reviews, comments, and tweets are available online and will keep being generated every day. It provides us good opportunities to enhance businesses based on these reviews or tweets in a way that cannot be achieved several years ago.

Due to the availability of user-generated text online, it is also witnessed that a lot of studies have been done that specifically focus on sentiment analysis on social media,

---

[1] http://jmcauley.ucsd.edu/data/amazon/
[2] https://www.ebayinc.com/our-company/who-we-are/
[3] https://s22.q4cdn.com/826641620/files/doc_financials/2018/q4/Q4-2018-Shareholder-Letter.pdf
[4] https://business.twitter.com
[5] https://cdn.cms-twdigitalassets.com/content/dam/business-twitter/resources/Customer_insights_2016.pdf

especially in Twitter by applying a variety of methods. Among these methods, we can see that lexicon based approach which is used to solve general sentiment analysis problems is introduced for Twitter sentiment analysis tasks; Supervised and semi-supervised learning based approach are two most popular methods that are used for Twitter sentiment analysis when at least a certain amount of training data is available; Also, there is a trend that different approaches are combined together for Twitter sentiment analysis in order to utilize the advantages from different methods, which makes hybrid approaches becoming more and more popular in recent years.

Although it is true that a big amount of works has been done in previous works, there are still some gaps to be filled. First of all, the availability of Twitter specific training datasets is becoming more and more limited for the past several years, and there is a lack of researches that using graphic emojis as noisy labels when building training datasets. Secondly, for the studies that were using noisy labeled training datasets which have no topic limitations, few attentions are paid to group the tweets based on their latent semantic meanings which will hurt the classification performance due to the existence of synonymy and polysemy. Furthermore, although it is important to introduce data processing details which can help to apply the proposed model to solve real business problems in a daily basis, few previous studies are paying enough attention to this part. Therefore, we think it is essential for us to propose a comprehensive sentiment analysis pipeline which includes data collection, sentiment classification, and a use case study to provide a mostly automated, configurable system that can be used by companies or institutions in a daily basis for free. Most importantly, each component in our proposed pipeline will fill the corresponding gaps mentioned above.

In this study, a novel topic-model based hybrid system is introduced, which aims

at solving the problems in Twitter sentiment analysis from end to end. We want to focus on the parts that have not been solved in previous studies in terms of data collection, Twitter sentiment classification, and data processing details when applying a model to solve real problems. For data collection, we propose a new automated way to collect training datasets based on graphic emojis which tries to fill the gap of currently there are few publicly available training datasets and the con of using string emoticons; For sentiment classification, we propose a topic-model based hybrid sentiment classification model which is can find the best topic model among three different topic models (LDA, HDP, and LSI) in an automated way by employing coherence score; A use case is also illustrated to explain the details about how to utilize this model for a real business problem in the perspective of data processing in a daily basis. The major goal of proposing the whole pipeline is to make most of the steps in the pipeline automated and configurable, to give it potential to be used by academics or small or medium-size companies for free.

1.2 Terminology used

In sentiment analysis related researches, it is not surprising that some terms have similar meanings could lead to confusion to readers. In this section, we will discuss the definition and scope of some commonly used words in sentiment analysis domain. Meanwhile, we want also to define some terms we use in this thesis which have specific meanings in the context of our study.

Sentiment, motion, and opinion. From [1], emotion is defined as a complex psychological state, which plays a key role in operating motivators, including happiness, sadness, anger, fear, surprise, and disgust, and so on. In contrast, sentiment refers to a mental attitude, which is created based on emotion and used to convey the thoughts of the individual deriving from his emotion. In other words, sentiment is like a bridge in the

middle of emotion and action, which is highly organized. All kinds of emotions can affect sentiment, which may lead to action later.

Based on Oxford dictionary, opinion is a view or judgment formed about something, not necessarily based on fact or knowledge. For example, as for a particular topic, different experts could have different opinions on it, which might contain sentiment or not. In some context, the same discourse could be regarded as sentiment or opinion, although there could be minor difference in some cases. Overall, an opinion is more of a person's view about something while sentiment is more about a feeling towards something.

In our study, our focus is the sentiment expressed from the given text instead of emotions or opinions.

Sentiment analysis, opinion mining, and subjectivity analysis. Sometimes, people interchangeably use sentiment analysis, opinion mining, and subjectivity analysis when referring to similar or same tasks. According to [2], opinion mining originates from the information retrieval community, and aims at extracting and further processing users' opinions about products, movies, or other entities, whereas sentiment analysis was initially formulated as a natural language processing task of retrieval of sentiments expressed in texts. While in [1], opinion mining and sentiment analysis are not significantly distinguished by all defined as "Sentiment analysis or opinion mining is the computational study of people's opinions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics and their attributes." In contrast, subjectivity analysis focuses more on "determine whether s is a subjective sentence or an objective sentence". Another definition of sentiment analysis could be found from [3]: "sentiment analysis refers to the general method to extract subjectivity and polarity from text (potentially also speech), and semantic orientation refers to the polarity and strength of words, phrases, or texts. Our

concern is primarily with the semantic orientation of texts, but we extract the sentiment of words and phrases towards that goal."

Based on all these definitions from previous study, we want to define the task and scope of sentiment analysis in our study as following:

"Sentiment analysis aims at extracting the sentiment orientation for the given text and assigning them to either positive or negative class. It also includes the techniques that could be potentially used to achieve this goal."

By using this definition, firstly we want to distinguish sentiment analysis as a different task from opinion mining and subjectivity analysis. In our thesis, sentiment analysis focuses on assigning positive or negative polarity to the given text, while opinion mining and subjectivity analysis focus more on detecting subjectivity from the text. Secondly, we want to clarify that based on our definition, sentiment analysis is a binary classification task rather than detecting the sentiment strength (one possible way is to assign a score from 1 to 5 based on the strength of the sentiment, or other score scopes, for both positive and negative). We do not consider detecting sentiment strength because of the following reasons: In order to detect the sentiment strength for the given text, training data with sentiment strength is required to contain strength as well which is hard to acquire in an automated way; An alternative could be using lexicon that has sentiment strength for each word, but lexicon-based approach performs poorly for Twitter sentiment analysis. Meanwhile, we want to compare our proposed method with previous works, and previous works mostly consider sentiment analysis as a 2-way or 3-way classification task as well. Overall, due to the goal of building an automated pipeline, and making comparisons with previous works, we want to keep our task as a binary classification problem.

Twitter sentiment analysis. As an extension of classic sentiment analysis tasks, Twitter sentiment analysis refers to the problem of analyzing the messages posted on Twitter in terms of the sentiments they express [4]. The speciality of Twitter's environment makes Twitter sentiment analysis a subtopic under general sentiment analysis task. Compared with sentiment analysis based on other text types, tweets have much shorter sentence, and contains a variety of slang, buzz words, misspelling, and Twitter specific features like URLs, mention, hashtag, emojis and so on. These traits need to be considered only in Twitter sentiment analysis tasks.

Hybrid or hybrid approach. Generally, the word "hybrid" or the phrase "hybrid approach" is from the previous works of Twitter sentiment analysis which refers to the method that combining two or more existing approaches together to build a new approach. The approaches that being combined could be from lexicon-based approach, the subcategories of machine learning approaches (supervised learning based approaches, semi-supervised learning based approaches, and unsupervised learning based approaches) but not limited to them.

In our thesis, when we are discussing our proposed sentiment classification model, we use the "hybrid" to refer to our method that combines topic modeling (unsupervised learning), clustering (unsupervised learning), ensemble learning and single classifier building (supervised learning) together. In the literature review section, "hybrid" is used as the general meaning.

Topic model or topic modeling. Topic modeling refers to the task that aims at uncovering the latent topics from a collection of texts which is firstly proposed in [5]. While topic model refers to the specific model that help to achieve this goal, like Latent Semantic Indexing (LSI), Probabilistic Latent Semantic Indexing (pLSI), Latent Dirichlet Allocation

(LDA), Hierarchical Dirichlet Process (HDP), and so on. There will be detailed explanations in Chapter 4 before we have an in-depth discussion of applying topic modeling in our approaches.

Pipeline or proposed pipeline. In the thesis, pipeline or proposed pipeline will refer to the combination of our proposed data collection method, sentiment analysis model, and data processing details illustrated by the use case in a sequential way which covers the majority components in a typical sentiment analysis workflow that could be used as a whole.

Sentiment classification. We have defined sentiment analysis refers to the task of extracting the sentiment orientation for the given text and assigning them to either positive or negative class, and also the techniques that could be potentially used to achieve this goal. It could contain a series of related tasks also before and after extracting the sentiment orientation. Therefore, we use the term sentiment classification to specifically refers to the core steps in a sentiment analysis task: data preparation, sentiment orientation extraction, and evaluation.

Comprehensive or comprehensive pipeline. The term comprehensive usually refers to including all or nearly all components or aspects of something. In a typical workflow of Twitter sentiment analysis, at a high level, it usually contains data collection, sentiment classification, and apply the model on a use case. Since our proposed pipeline covers these three components in the thesis, we will use the word "comprehensive" to describe the trait of our pipeline that has a nearly full coverage of essential components in a Twitter sentiment analysis workflow.

Noisy labels. In Twitter sentiment analysis studies, there are two commonly used ways to label training datasets in terms of their sentiment orientation. The first way is to

label the datasets with human efforts. One or more human annotators will participant the labeling process, and if there is more than one person, their opinions on the annotations can be considered based on majority voting or based on some other schema. The other way to label the tweets is using some existing features which are called noisy labels in Twitter to categorize them into different polarities. The most commonly used one is emoji or emoticon, which is considered indicators for the sentiment of the related Twitter. Since there are no human judgements involved, and there could be noise by annotating the tweets this way so that these emojis or emoticons are called "noisy labels".

1.3 Motivation

It is true that a lot of works have been done for Twitter sentiment analysis for the past several years. While after being inspired by Prophet[6], a time series forecasting package built by Facebook that aims at making forecasting at scale when solving real forecasting problems, we propose that to apply a Twitter sentiment analysis model to solve real sentiment analysis problems on Twitter, the model should be mostly automated, configurable, can be updated periodically, and include the major parts of sentiment analysis in the pipeline to keep it consistent. Making the model automated aligns with the main idea of software engineering which can reduce human effort and the probability of errors caused by human actions; Making the model configurable is also important because in the real use cases, models that easy to tune and configure are popular than the ones more like black boxes; Another thing that few previous works have discussed is how to make the model run periodically and always up-to-date, which is important when building online tools; And few works have talked about all the major

[6] https://facebook.github.io/prophet/

components of a Twitter sentiment analysis pipeline in one study. In our study, the design of the data collection method is an essential step to make the whole pipeline automated; The design of sentiment classification model is to ensure that the components within the model are configurable; The use case aims to show how to run our pipeline online with continuously coming training data and test datasets and update the model periodically and always choose the best model for usage. By doing this, we want to make our proposed method applicable for real business problems.

1.4 Contributions

Based on the problems statements in the background section, the major goal of this thesis is to propose a comprehensive pipeline for Twitter sentiment analysis that is mostly automated and configurable, so that it can be utilized to solve real business problems in a daily basis and benefit companies and institutions as a free tool. Specifically, the following contributions are made:

- Propose a new method to collect and build Twitter training datasets based on manageable and traceable graphic emojis in an automated way, without being impacted by Twitter's redistribution policy and time effect;
- Propose a novel topic-model based hybrid sentiment analysis model to consume Twitter training datasets annotated by noisy labels by utilizing unsupervised learning and supervised learning approaches;
- Propose an automated way to find out the best topic model without human effort when applying several different topic models with different parameters setting;
- Provide data processing details when applying our proposed pipeline to solve a real business problem on a daily basis.

- Propose a comprehensive pipeline including training data collection, Twitter sentiment classification, and data processing details for a real use case which is mostly automated and configurable.

1.5 Thesis Overview

In Chapter 1, the background and introduction of sentiment analysis and Twitter sentiment analysis will be discussed along with the motivation, current gaps, and problems, contributions at a high level. Since we aim at building a Twitter sentiment analysis pipeline, literature review about existing Twitter sentiment datasets, Twitter sentiment analysis approaches, and related use cases and tools are reviewed respectively in different sections of Chapter 2, which are followed by literature summaries and gap analysis in each section. At the end of the Chapter, a problem statement section summarizes the gaps in the entire pipeline and clarify what problem we are trying to solve. In Chapter 3, our proposed data collection method is described in detail, which is followed by Chapter 4, a novel sentiment classification model is proposed based on topic modeling, clustering, and supervised learning. Chapter 5 is used to shows the evaluation of our proposed sentiment classification model in terms of classification results and efficiency. A use case can be found in Chapter 6 to introduce the details about applying our proposed pipeline to solve a real problem on a daily basis followed by the review and discussion of the whole pipeline. In Chapter 7, we summarize our works in conclusion and discuss future works.

Chapter 2: Literature Review

A typical workflow or pipeline of Twitter sentiment analysis is shown in Illustration 1, which usually start with data collection, followed by data preparation, sentiment orientation extraction and evaluation, and also include the use cases or tools built on top of the previous steps. In this chapter, we will review the steps in this workflow in terms of data collections, sentiment classification, and use cases or tools building, respectively.

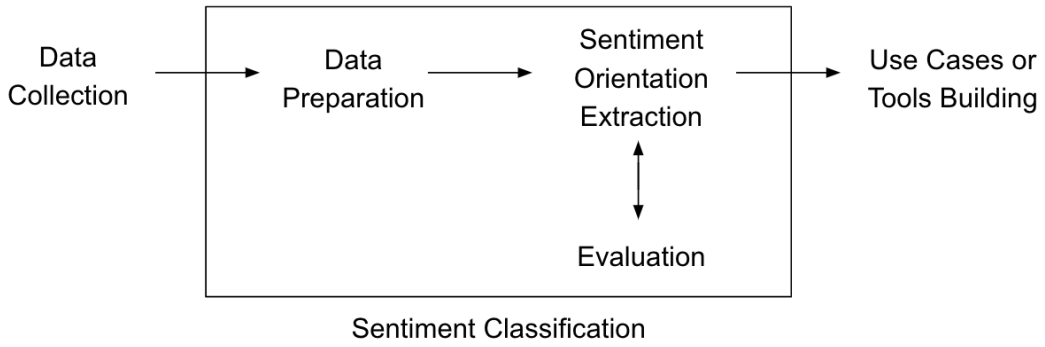


Illustration 1 A Typical Workflow of Twitter Sentiment Analysis

2.1 Existing Dataset for Twitter Sentiment Analysis

With the increasing need of Twitter sentiment analysis, a variety of Twitter specific datasets are built by different researchers or workshops. In this part, Twitter specific datasets that are commonly used in previous works are reviewed.

2.1.1 Existing Twitter Datasets review

Sanders Twitter Sentiment Corpus

Sanders Twitter Sentiment Corpus[7] is a Twitter dataset that consists of 5512 instances towards four topics: Apple, Google, Microsoft, and Twitter. For each tweet, it is

[7] http://www.sananalytics.com/lab

classified into one of the 4 following categories: positive, negative, neutral, irrelevant. With a number 570 positive, 654 negative, 2503 neutral, and 1786 irrelevant tweets, we can see that the instances that could be used for model training might be smaller than the actual size of the corpus since irrelevant tweets account for a big portion. This dataset is used in studies [6], [7] and others. It could be a helpful dataset when your study is towards the above-mentioned four brands or other related brands and domains. Currently this dataset requires to be downloaded online[8] since the directly distribution is limited due to Twitter's policy.

Obama-McCain Debate (OMD)

Obama-McCain Debate Dataset [8] is a Twitter dataset is built with the motivation of getting a better understanding of social media events. It consists of 3238 tweets about the first presidential TV debate between Barack Obama and John McCain. Collected in September 2008, each of the tweets is classified into positive, negative, mixed, and others categorized by Amazon Mechanical Turk. The dataset also keeps the rating from human annotators so that the researchers who want to use this dataset can decide how to get a subset based on the rating. This dataset is used in studies [7], [9], and others. The major contribution of this dataset is it provides a dataset for a controversial topic which contains complex opinions and language style than other daily topics. It could be helpful when analyzing political related topics in Twitter or being used as a test dataset to evaluate the performance of the model when it comes to complex political topic. While the collected data is more than ten years ago, so it could be outdated in some cases.

[8] https://github.com/zfz/twitter_corpus

Health care reform (HCR)

Health care reform (HCR) dataset [10] is collected about the health care reform in the US, and a subset of this corpus is labeled with a total of 2516 tweets including the training, development, and test components. Each tweet in this strongly debated topic is categorized into one of the following four categories: positive, negative, neutral, irrelevant. Although the author also tries to label the sentiment polarity for different topics (health care reform, Obama, Democrats, Republicans, Tea Party, conservatives, liberals, and Stupak), the sentiment orientation in topic level is the same as the ones in message level. This dataset is used in studies [7], [11], [12], and other researches. As another popular Twitter dataset, it contributes the same ways as Obama-McCain Debate Dataset. But similarly, it also might suffer from time effect since it is released nine years ago.

Stanford Twitter Sentiment (STS)

Stanford Twitter Sentiment (STS) dataset is created in [13] by graduate students at Stanford. This dataset has 1.6 million instances crawled by Twitter API with a filter of the ones that contain positive or negative string emoticons. The sentiment labels of the tweets are also based on emoticons which means the ones containing :), :-), :), :D, or =) within the tweet would be auto-annotated into positive category while the ones having :(, :-(, or : (would be auto-labeled as negative category. In the end, 0.8 million positive tweets and 0.8 million negative ones are stored in the dataset evenly. Since the tweets are collected based on emoticons, so there are no specific topics covered in this dataset. The pro of this dataset provides a fruitful source of tweets that is labeled based on noisy labels; Also, it shows the potential to utilize nearly cost-free, around 40000 tweets being generated every day; While one of the cons is the way to utilize noisy labels could be revised nowadays since graphic emojis are more popular, manageable, and traceable than string emoticons;

In addition, the released date of this dataset is 10 years ago so that building models on top of it might be impacted by time effect.

Edinburgh Twitter Corpus (ETC)

Edinburgh Twitter Corpus is firstly released in 2010 which was collected and organized by Petrovic et al. [14]. The corpus has 97 million tweets collected by Twitter streaming API from November 2009 to February 2010. With about 100 million data, there is also no limitation of the topics since the trait of Twitter streaming API (a random subset of all the tweets would be returned). This dataset was used in [15], [16]. Compared with other existing datasets, the volume of this dataset is quite big which allows the researchers to have a sufficient data source for building a series of applications. While the limitation is that there is no annotation in terms of sentiments so that further work needs to be done before it is utilized for sentiment analysis tasks. Unfortunately, this dataset is no longer available publicly.

O'Connor's Corpus (OC)

O'Connor's Corpus is firstly proposed in [17]. In this dataset, around 1 billion tweets are collected during 2008 and 2009 motivated by need of getting a better understanding of public opinion towards polling. The data is captured by Twitter API and "Gardenhose" real-time stream with a range of 100k to 7 million tweets per day. This dataset is also used in [18]. Sharing some similarity with Edinburgh Twitter Corpus, this corpus also has millions of tweets which makes it a better representative of the overall tweets; However, it is not annotated either so that the researchers need to go one step further to find a way to label the sentiment polarity themselves which could be computationally intensive in terms of the size of tweets of this dataset.

Dialogue Earth Twitter Corpus (DETC)

Dialogue Earth Twitter Corpus (DETC) dataset is a part of Dialogue Earth Project[9] which consists of three datasets: WA, WB that are about weather with 4490 and 8850 tweets and GASO which is about gas price that having 12770 tweets. The tweets are labeled by human judgments and each tweet is classified as one of the following labels: positive, negative, neutral, and not related to the target topic. It is used in study [19]. This dataset is domain-specific which could be a benefit for related studies. Since this project is not currently active, the dataset is not publicly available now.

SentiStrength Dataset (SS)

SentiStrength dataset is proposed firstly in [19] which is designed as a test dataset to evaluate the performance of SentiStrength, a lexicon-based classification model. It is not a Twitter-specific dataset, but it has a subset which is collected from Twitter and also be used by researches when solving Twitter sentiment classification tasks. SentiStrength dataset consists of six data sets (BBC Forum posts, Digg.com posts, Runners World forum posts, Twitter posts, YouTube comments) that labeled by human judgments and a combined data set containing all of the six datasets. With a total of 4242 tweets, each of them is annotated in terms of sentiment strength between -1 (not negative) and -5 (extremely negative) and between 1 (not positive) and 5 (extremely positive). This dataset could be used for performance evaluation especially when comparing the performance of the proposed model on the texts from different channels.

[9] http://www.dialogueearth.org

STS-Gold Dataset (STS-Gold)

STS-Gold Dataset is proposed in [20] with the motivation to create a new dataset where tweets and entities are annotated independently for more fine-grain analysis. The dataset is built by the following steps: Firstly, select a 180k subset of Stanford - Twitter Sentiment Corpus; Secondly, extract entities by using Alchemy API to draw the semantic concept class and keep; In addition, keep only the top 2 entities for each concept. All the tweets in the dataset are classified into five classes: negative, positive, neutral, mixed, and other by a group of graduate students. For both message and entity level, only the tweets that three annotators agreed would be kept into the dataset. This dataset is used in [21], [22]. The major contribution is that it is a Twitter dataset that contains both message and entity level sentiment annotations.

SemEval Datasets

The SemEval (Semantic Evaluation) is a series of workshops that focus on the evaluation of semantic analysis system [10] which evolves from the Senseval word sense evaluation series[11]. In 1998, the first workshop was held in England and mainly focused on Word Sense Disambiguation. Until now, there have been 13 workshops held, and the latest one is SemEval-2019 and the latest one has Twitter sentiment analysis task is SemEval-2017.

For each workshop, a variety of tasks would be set, and the related dataset would be provided for participants' usage. From time to time, Twitter sentiment analysis was set as part of the tasks, which also brings several Twitter-specific sentiment analysis datasets

[10] https://www.cs.york.ac.uk/semeval-2013/
[11] https://en.wikipedia.org/wiki/SemEval

for researchers' usage even after the workshop.

SemEval-2013 Task 2 Dataset

SemEval-2013 task 2 Dataset[12] is built for the SemEval-2013 task 2 and released in 2013. The topic coverage of the dataset consists a mixture of entities (e.g., Gadafi, Steve Jobs), products (e.g., kindle, android phone), and events (e.g., Japan earthquake, NHL playoffs). Specifically, two sub-datasets were created for the aspect level task and message level task.

For the expression level task, the collected tweets are labeled into three classes: positive, negative, and objective with a total number of 26k; For the message level task, the collected tweets are also divided into three classes: positive, negative, and objective-OR-neutral with a 6.7k in total. The dataset is still available online and used in [23], [24], [25], [26], [27], [28] and other researches.

SemEval-2014 Task 9 Dataset

SemEval-2014 task 9 dataset[13] is a rerun based on SemEval-2013 task 2. It shares the same training and development dataset as SemEval-2013 task 2 while the test dataset is different which consists of Twitter-2013, SMS-2013, Twitter-2014, Twitter-sarcasm-2014 and LiveJournal-2014. This dataset is used in [29], [30], [31], and other researches.

SemEval-2015 Task 10 Dataset

SemEval-2015 task 10 Dataset[14] is an expanded version based on SemEval-2013 task 2 Dataset and SemEval-2014 task 9 dataset. Four subtasks are set under SemEval-2015 task 10 (an expression-level subtask, a message-level subtask, a topic-related subtask,

---

[12] https://www.cs.york.ac.uk/semeval-2013/task2/index.php%3Fid=data.html
[13] http://alt.qcri.org/semeval2014/task9/index.php?id=data-and-tools
[14] http://alt.qcri.org/semeval2015/task10/index.php?id=data-and-tools

a trend subtask), and four datasets are provided correspondingly. For subtask A and B, the training datasets provided are the same as SemEval-2013 task 2 and the test datasets are the same as SemEval-2013 task 2 and from SemEval-2014 Task 9; For task C and D, the dataset consists of 486 tweets which are a combination of around 10 tweets per topic for 44 topics. This dataset is used in [32], [33], and other researches.

SemEval-2016 Task 4 Dataset

SemEval-2016 task 4 dataset[15] is a partially rerun based on SemEval-2015 task 10 and an extended to some new tasks. The new parts in task of 2016 is replacing classification with quantification and replacing the standard two-point scale (positive / negative) or three-point scale (positive + neutral + negative) with a five-point scale (very positive + positive + ok + negative + very negative). Respectively, the datasets for task C, D, E share the same new dataset with a five-point scale labeling in terms of the sentiment score towards the topic provided by the organizers.

SemEval-2017 Task 4 Dataset

SemEval-2017 task 4 dataset[16] also inherits the major task for the last year but applied some changes. Firstly, Arabic is added as a new language, and also information about Twitter users is added. For training datasets, a list of datasets from previous years are provided and could be freely used in the tasks while the new datasets for Arabic are separately provided. It is used in [34] and other researches.

2.1.2 Literature summary

A comparative table is shown in Table 1 in terms of the name of the dataset,

[15] http://alt.qcri.org/semeval2016/task4/index.php?id=data-and-tools
[16] http://alt.qcri.org/semeval2017/task4/index.php?id=data-and-tools

released year, classes labeled in the dataset, language used of tweets, analysis level, the volume of tweets, related topics, acquired method, and annotated method. We can see that a tweet could be classified into either of the following labels: positive, negative, neutral, mixed, others, irrelevant, subjective or even into a score scope from -5 (most negative) to +5 (most positive). In another perspective, the datasets are created for different analysis level including message level, entity level, and topic level. Message level datasets provide the sentiment polarity labels for the entire message while entity level datasets provide labels for sentiment towards some specific entities within the tweets. The first two levels exist for generic sentiment analysis tasks while topic level is a Twitter-specific level which refers to label the tweets based on the provided topics in the tweets. Topic could be an entity or something else, like, based a hashtag or a meme on Twitter. We separate this apart since some SemEval datasets provide topic level labeled datasets.

When it comes to a domain-specific dataset, we can see most of the human-labeled datasets are built based on Twitter Search API[17], and the tweets are collected based on hashtags, mentions, and keywords. Therefore, they always cover some particular domains based on their ways of searching. Some big datasets that built by using noisy labels are more generic and don't have a specific domain coverage in the dataset which are collected by Twitter Streaming API[18].

It is clear that how researches can benefit from this series of sentiment analysis corpus for Twitter afterward. Firstly, they are publicly available which provides good resources for model building and facilitates the comparison across different models; Also,

[17] https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.html
[18] https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data.html

Table 1 Literature Summary of Sentiment Analysis Datasets

Dataset Name	released year	classes	language	level	volume in training dataset	related domain	acquired way	annotation
Sanders Corpus Dataset	-	positive, negative, neutral, irrelevant	English	message	5512	four different topics (Apple, Google, Microsoft, Twitter).	hashtag and mention	Human judgment
Obama-McCain Debate (OMD)	2008	positive, negative, mixed, others	English	message	3238	first U.S. presidential TV debate	-	Amazon Mechanical Turk
Health care reform (HCR)	2010	positive, negative, neutral, irrelevant	English	message/ entity	2516	health care reform in the USA	hashtag	-
Stanford Twitter Sentiment (STS)	2009	positive, negative	English	message	1.6 million	no limitations	string emoticons	Based on emoticons
Edinburgh Twitter Corpus (ETC)	2010	-	No limitation	-	97 million	no limitations	Twitter streaming API	-
O'Connor's Corpus (OC)	2010	-	No limitation	-	1 billion	no limitations	Twitter API, "Gardenhose" real-time stream	-
Dialogue Earth Twitter Corpus (DETC)	before 2012	positive, negative, neutral, not related to the target topic	English	message	26110	weather and gas price	-	Human judgment
SentiStrength Dataset (SS)	2012	between -1 (not negative) and -5 (extremely negative) and 1 (not positive) and 5 (extremely	English	message	4242	BBC Forum posts, Digg.com posts, Runners World forum posts, Twitter posts, YouTube comments	-	Human judgment

		positive)						
STS-Gold Dataset (STS-Gold)	2013	negative, positive, neutral, mixed, other	English	message/ entity	2111	tweets with top entities	based on existing dataset	Tweenator
SemEval-2013 task 2 - A SemEval-2014 task 9 - A SemEval-2015 task 10 - A	2013	positive, negative, objective	English	entity	26462	a mixture of entities (e.g., Gadafi, Steve Jobs), products (e.g., kindle, android phone), and events (e.g., Japan earthquake, NHL playoffs).	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2013 task 2 - B SemEval-2014 task 9 - B SemEval-2015 task 10 - B	2013	positive, negative, objective-OR-neutral	English	message	6756	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2015 task 10 - C SemEval-2015 task 10 - D	2015	positive, negative, neutral	English	message/ topic	486	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2016 task 4 - A	2016	positive, negative, neutral	English	message	6000	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2016 task 4 - B/D	2016	positive, negative	English	topic	4346	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2016 task 4 - C/E	2016	positive, negative, neutral	English	topic	6000	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2017 task 4 - A	2017	positive, negative, neutral	Arabic	message	2684	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk
SemEval-2017 task	2017	positive,	Arabic	topic	1324	Same as above	Keywords and	CrowdFlower

4 - B/D		negative					twitter hashtags	or Mechanical Turk
SemEval-2017 task 4 - C/E	2017	positive, negative, neutral	Arabic	topic	2682	Same as above	Keywords and twitter hashtags	CrowdFlower or Mechanical Turk

we can see the efforts that were put into providing different volumes or annotation levels of datasets over years.

2.1.3 Gap analysis for existing Twitter datasets

Although there are some high-quality datasets like the ones provided by SemEval workshops, the availability of training dataset is still quite limited. And there are some gaps we can see after reviewing the existing datasets:

Firstly, with the change of Twitter's policy and the consideration of user privacy for the past five years, there are much more restrictions about redistributing tweets which results in a lack of publicly available Twitter datasets currently. To be more specific, due to Twitter's most recent published policy which started to be effective on November 3, 2017[19], republishing tweets requires the user's express consent for privacy protection consideration. While if you have a dataset with thousands of tweets posted by different users, it would be different to get all their consents or even get in touch with them. Therefore, the availability of Twitter specific datasets becomes less and less due to the stricter policy of Twitter for republishing.

Secondly, time effect is a problem, especially when it comes to datasets for Twitter sentiment analysis. One of the most different features that Twitter has compared with other text types is the special language style people use and the way people generate buzz words or hashtag or some other new words or phrases every day, which is not a feature for customer reviews, blogs, or news. Therefore, if a training dataset for Twitter is created 5 years ago and being used today to build a model that tends to be applied on the test data

[19] https://developer.twitter.com/en/developer-terms/policy.html

drawn yesterday (assuming that in the business environment, we always want to know the sentiment orientation or trend for the most recent tweets acquired), it can be problematic because the model never get an opportunity to learn the most recent buzz words or hashtag from the training dataset since it was built 5 years ago. Overall, the time effect is a problem that needs to be considered when building Twitter sentiment analysis models.

Thirdly, nearly all studies collecting their own data previous only consider string emoticons as noisy labels instead of graphic emojis which are easier to be traced and more manageable. After we discussed the first two gaps, it looks collecting your own dataset with noisy labels is a doable option. By doing it, you will not rely heavily on existing datasets published by other researchers, and also can solve the time effect issue by keeping collecting streaming tweets. While one concern is the previous works, only use string emoticons as noisy labels, which are harder to be traced, and managed compared with graphic emojis. When using string emoticons, there is no rule of how to write an emoticon so that people can make different combinations and the researchers need to use complex regular expression to match them, which makes it hard to be traced; Meanwhile, it is unrealistic to build an exhausted list of all the string emoticons that have ever been used but only the ones the researchers have seen which makes them hard to manage. These problems can be solved by using graphic emojis instead because the graphic emojis has their own Unicode, which makes it easy to trace and manage.

Based the above gaps, it is essential to create a data collection method which can be used to build training dataset of Twitter sentiment analysis which has few restrictions, can evolve over time, and utilize better noisy labels like graphic emojis.

2.2 Twitter Sentiment Analysis Approaches

In this section, we reviewed 48 papers related to Twitter sentiment analysis and focus on the approaches they used. The reviewed papers are categorized into three major categories (lexicon based approach, machine learning based approach, and hybrid approach), and several subcategories for categorization. After reviewing, a summary of the literatures and the gaps are discussed in this section as well.

2.2.1 Lexicon-based approaches

Lexicon based approach refers to the general method that using a lexicon which contains a variety of sentiment words that are manually labeled by domain experts or linguists to facilitates the process of determining sentiment polarity of the given text. The best features of the lexicon-based method include its simplicity which makes the result explainable, and the potential to be integrated on different applications easily without intensive storage and computational resources required. We can see some well-known lexicon are still widely being used in researches nowadays like Bing Liu's Lexicon [35], Sentiment Strength [36], a series of research about SentiWordNet [37], [38], NRC Emotion Lexicon [39], MPQA Subjectivity Lexicon [40], Sentiment140 Lexicon[20], and so on.

A research that uses lexicon as their primary method for extracting sentiment polarity was done in [3]. They proposed a system called Semantic Orientation CALculator that utilizes several features including Adjectives, Nouns, Verbs, and Adverbs, Intensification, Negation features, Text-Level Features which extend to other parts of speech compared with previous works. The system achieved a 78.74% accuracy which

[20] http://help.sentiment140.com/

proved that the method is robust and can result in excellent cross-domain performance.

The pro of this paper is that it made a thorough comparison about the strength and weakness about lexicon-based methods, supervised learning based method, and semi-supervised learning based method which justifies the value of their proposed lexicon based approach and the scenarios that it will perform best; For tweets sentiment analysis, we could refer the ways that lexicon-based approach, like, how it deals with negation and intensification. But it is hard to transplant lexicon methods directly on tweets sentiment classification tasks since the language style are quite different. Meanwhile, the words used in tweets tend to evolve over time, but customers reviews are relatively static.

Another work which relies on the lexicon-based method to deal with aspect level sentiment analysis task is [29]. Their system worked as a pipeline of aspect extraction, parsing aspects from reviews, sentiment classification based on lexicon and category identification based on the aspects for each review by utilizing the combinations of the Bing Liu Lexicon [35], MPQA [40] and SentiWordNet [37]. The best F-measure achieved 0.808 when applying the proposed model on test data from SemEval 2014 Task 4 about restaurants and laptops.

In the paper, they proposed a complete system for aspect level analysis, but they didn't go deep into the implicit aspects and also the handling of negation.

Although lexicon-based approach already achieved some great results in different tasks, when it comes to social networks, it is not the most applicable option which may lead to undesirable performances due to the intensive use of informal language, slangs, and buzz words.

In paper [41], this problem acquired the awareness of the authors. They proposed a

framework which was a combination of a slang classifier, an emoticon classifier, a SentiWordNet based classifier, and a domain-specific classifier in a sequential way. The slang classifier and the emoticon classifier would be used first to examine the sentiment polarity of the tweets, while the SentiWordNet based classifier was applied for general classification of tweets, and the domain-specific classifier was used to correct the domain specific terms not incorporated in SentiWordNet or incorrectly categorized in previous classifier.

The first pro of this paper is the construction of a slang dictionary, including 8453 slangs with their meanings, which was built based on four online sources and the help of human annotations. Moreover, an emoticon dictionary was also created with 721 emoticons included. Also, word sense disambiguation was applied by aggregating the sentiment scores for each part-of-speech of that word. Last but not least, a new way of detecting domain-specific words was proposed and used in this study.

One concern is the emoticon dictionary only contains string emoticons but not graphic emojis which are commonly used in Tweets.

Instead of regarding sentiment classification as a 2-way or 3-way classification, some research also considers it as a more fine-grain task which can potentially provide more insight to the stakeholders. In [42], the authors tried to response the increasing interest in the affective dimension of the social web especially in Twitter. To fill the gap that most sentiment analysis algorithms were not ideal to this task because they utilize indirect sentiment indicators, they wanted to test an improved version of the SentiStrength for sentiment strength detection based on direct sentiment indicators. After testing with six social media datasets including Twitter, they found that SentiStrength not always

outperform machine learning based approach, especially when it comes to detect the positive post in news; While SentiStrength did outperform baseline accuracy for positive class for all datasets and also the baseline for negative class expect for 2 datasets, which showed some extent of robustness across different datasets with different language type and style.

One major contribution is that the authors applied a fine-grain sentiment analysis to response the increasing interest of it. Compared with 2-way or 3-way classification tasks which categorized the given text into positive or negative class, this task focused on assigning sentiment strength score from 1 (weakest) to 5 (strongest) for both positive and negative classes. They also tried to verify the robustness by applying SentiStrength on social network related datasets from 6 sources.

It would be better if the authors could test their SentiStrength based approach on social media datasets that built on different times, since the changing language style and the newly created hashtags, buzzwords, and so on is one of the most special features compared with other text style. Therefore, it would be good to know if the performance of their proposed solution will still be robust over time.

2.2.2 Machine learning based approaches

Machine learning based approach is one of the most popular methods for Twitter sentiment analysis nowadays [4]. It defines sentiment analysis as a classification problem by utilizing some well-known machine learning algorithms and choosing specific linguistic related features to categorize the given text into different classes.

Machine learning based approach is also a pretty generic category in sentiment analysis. To be more specific, it includes supervised learning based approach, semi-

supervised learning based approach, and unsupervised learning based approach. supervised learning based approach tries to take advantage of a number of labeled training data while the unsupervised is preferred when these training data is not available. Semi-supervised is in the between of these two approaches: when there is a limited amount of training data available but a high volume of unlabeled data which might help to enhance the training dataset. All these three kinds of machine learning approaches are widely used for Twitter sentiment analysis tasks. In the following sections, we will review the works have been done for each method, and also summarize the current status and gap in this domain.

2.2.2.1 Supervised learning based approaches

The supervised learning approach is employed in sentiment analysis research when a variety of high-quality labeled training data is available. The most popular algorithms that being used in supervised learning based approaches include support vector machine, logistic regression, Naïve Bayes, decision tree, and ensemble learning algorithms and convolutional neutral network.

Single classifier based approaches

In [43], the authors aimed at solving the problem of what do people think about the product and what the corresponding sentiments are towards it. They utilized Twitter API to acquire the data they need for building the corpus and explain how to train a classifier for sentiment analysis based on the corpus the built. By using emoticons from tweets as positive and negative labels, they turned to some newspaper accounts to crawl some extra tweets to build neutral records. Interestingly, after they tried a group of algorithms, they found Naive Bayes gets the best results over SVM and CRF.

The pro of this paper is they tried to find a way to build a training dataset for all the

positive, negative, and neutral instances without human efforts. This could provide a way for people to utilize as much data as possible. In addition, they analyze the traits for positive, negative and neutral dataset and summarize some features that could help to indicate the sentiment polarity which is helpful for others to understand better about which features would work best in similar tasks.

While it would be better if they could clarify the volume of tweets for the training dataset. Also, the assumption that the tweets from newspaper accounts tend to be neutral is still in debate since they could also convey some paragraphs with sentiments.

One of the most popular algorithms being used in Sentiment Classification problems is SVM. In [44], three types of models were tested in their work including unigram model, a feature-based model, and a tree kernel-based model were being used to train the SVM model. Among the three models, unigram model was used as a baseline, which had an over 20% improvement over randomly guessing. While the feature-based model got a similar accuracy as the unigram model does by only using 100 features. Last but not least, the tree kernel-based model, a newly designed tree representation for tweets, performed best by achieving an accuracy of 75.39 on their test dataset.

The major contribution of this paper is the feature-based model performed comparably when using only 100 features and a smaller standard deviation in evaluation while the unigram counterpart requires more than 10000 features and having a bigger standard deviation. This could be a helpful dimension reduction when dealing with text classification tasks.

On the other hand, we can see that there were 11875 tweets collected without restriction of language, location, or anything else used by the authors, and there is no proof

that when applying the same method on other Twitter datasets the performance would keep stable or not.

[23] is a response to SemEval-2013 task 2, which trained two SVM classifiers to deal with a message-level task and a term-level task and ranked top in both of them. A variety of features were being used in this work including surface-form, semantic, and sentiment features. At the same time, the NRC Hashtag Sentiment Lexicon was generated, and Sentiment140 Lexicon was utilized as well. The proposed method achieved a 69.02 in F-measure for the message-level task and 88.93 for the term-level task.

[24] is the extended version of [23] that published one year later. In this version, another two lexicons Affirmative Context and Negated Context Lexicons were introduced and explained which made the model utilizes four lexicons in total. One improvement is that in their study the way to handle negation is quite novel in supervised learning based approaches. Also, the way they calculate the sentiment scores in lexicons got updated. With their new approach, an F-measure of 70.45 in message-level and 89.50 in term-level were witnessed.

We can see that these two studies utilized as many resources (in both corpus and computation level) as possible for achieving a better performance. For each tweet, a variety of features would be extracted from it and also two lexicons with more than 50k and 60k unigrams and more than 300k and 600k bigrams respectively were used. In addition, SVM is known for its high performance in text classification but also computationally expensive. Therefore, it might be hard for organizations to duplicate the performance achieved by this approach. Also, it would be better if the authors show the training time for creating the lexicons and building the model which might help people to estimate the performance in

business cases.

Some works utilize a supervised learning approach to deal with real-time processing on Twitter. The problem is the continuously incoming tweets and the difficulty to run the complex algorithm for each tweet which leads to low accuracy. In [6], several supervised methods were tested. The tweets for training are represented as n-grams while only the top n features were selected to improve the performance. Meanwhile, six feature selection methods including Chi-Squared, Filtered Feature, Gain Ratio, Info Gain, One R, and Relief were tested with the combination of three kinds of Perceptron classifiers. The best performance achieved an F-measure of 85 for subjectivity detection and 78 for sentiment analysis.

One of the highlights is although this is a 3-wayclassification problem, they treated a 3 classes identification problem as two 2 classes identification problems by running a subjectivity/objectivity classification task first and followed by a positive/negative classification task for those were distinguished as subjective in the first task. Another highlight is they consider how processing time varied with the number of features and try to test different feature reduction methods which is helpful for organizations that might not have many computational resources.

While it would be better for this study to try out different ensemble learning approaches. Apart from it, their training data was human-annotated, but their goal is to deal with streaming data. So that it's still unclear how to make the model evolves over time and achieve a comparable result on other domains other than the four topics (Apple, Google, Microsoft, and Twitter).

In [45], the authors proposed that the current commonly used features like tf-idf,

N-gram, and word embedding based vector representation perform badly when capturing the sentiment information. Instead, a quantum-inspired sentiment representation (QSR) model was proposed to solve this problem with the following steps: Firstly, extract sentiment phrases based on adjectives and adverbs which were considered as good indicators of subjective expression; Furthermore, single words and phrases extracted from the previous were modeled as a collection of projectors and encapsulated in density matrices through maximum likelihood estimation.

The highlight of this study is a new text representation was proposed based on quantum probability theory, which has been tested for information retrieval tasks but rarely used for sentiment analysis. Also, the way how to extract sentiment phrases from documents was also inspiring for our studies. Furthermore, case studies were used to show the effectiveness of the proposed approach.

One concern is in the experiment with OMD Twitter dataset, the computational time for the proposed QSR method was the longest one (1575.3 s using RF, 2821.2 s using SVM and 860 s using NB) even compared with standard CNN and FCDNN (81.5 s and 79.8 s), but the best result achieved when applying AT-LSTM method. Therefore, there is a possibility that the proposed solution suffers a longer running time without the most satisfying result.

Ensemble learning based approaches

After exploring the potential of individual algorithms, a lot of researches turn to ensemble learner to enhance the performance based on individual learners which lead to an increasing trend of the usage of ensemble learning in sentiment analysis. Different kinds of ways for ensemble learning are used including assembling different algorithms,

ensemble the same algorithms with different parameters, by employing different ensemble method like majority voting, bagging, boosting, stacking, etc. Based on the theory of ensemble learning, several weak learners could be assembled into a strong classifier, so we can see that some popular algorithms used in an ensemble classifier are built based on Random Forest, Bayesian Network, etc.

In [46], the authors compared the performance among two popular ensemble techniques: bagging and boosting (AdaBoost-r) and several individual learners including K Nearest Neighbors (KNN, k=5), two types of C4.5, Support Vector Machines (SVM), Multilayer Perceptron (MLP), Radial Basis Function Network (RBF), and Logistic Regression (LR) algorithms based on sentiment140 corpus in which subsets were extract as dataset with emoji labels and another one with manually labeled data. The result showed that ensemble learners achieved better performance on both datasets and bagging outperformed other models when data quality was a concern.

This study is a classic usage of ensemble method on Tweets, and it's good to see the ensemble approach outperforms the traditional method when data quality is a problem.

However, it is unclear that using sentiment 140 as test dataset is a good practice or not since it is labeled based on emoticons other than human judgments.

A similar research is [46] in which ensemble learning approach was also applied on Twitter sentiment analysis tasks. The authors elaborated on the process of data preprocessing, Feature Representation, Sentiment Classification using base classifiers.

The highlight of this study is that they proposed a new algorithm for ensemble learners. First, each learner was trained and make prediction individually on test data. Then the probability of the sentiment polarity for each tweet was calculated based on the results

from all the learners. After that, the learner's weights were evaluated by the accuracy they achieved divided by the summary of the accuracy's overall learners. Last but not least, the probability for each tweet was adjusted by integrating the weights from the learners and get the final probability score. The sentiment polarity of a specific tweet was decided by comparing the probability base on both the positive and negative scores. If the scores equaled to each other, then the score was determined by the tweet which has the highest cosine similarity value. In the experiment, the proposed ensemble classifier outperformed the individual learners and majority voting approach.

While a concern would be how scalable is this new approach compared with traditional methods in different scenarios on different datasets. If this question could be answered, the use case of this approach would be much wider.

In [7], the author applied ensemble learning to deal with Twitter sentiment analysis. The bag-of-word and feature hashing were tested as feature representation methods. From the research, they selected the method to vary the base classifiers or ensemble strategies to build the ensemble learner on topic of four base learners (Random Forest, Support Vector Machines (SVM), Multinomial Naive Bayes, and Logistic Regression). The results on a variety of public tweet datasets showed that classifier ensembles did improve the accuracies. The feature hashing provided worse results than the Bag-of-Words model in most of the datasets while it enabled a significant reduction in dimensionality.

One highlight is that feature hashing was used as a relatively new feature selection technique which produced features represented as hash integers rather than strings in traditional research. It was utilized to deal with the high dimensional input space of tweets and the sparsity problem, although in some cases, it would decrease the performance a bit.

One concern for this study is that based learners were supposed to be weak and unstable learners. While classifiers like Random Forest and SVM are usually considered to be strong and stable learners. It would be good for the author to justify the way they choose the base learners.

In [47], the authors focused on the domain of airline service and built an ensemble classifier for sentiment analysis on it. To solve the data imbalance problem, they run a random resampling to get a dataset with balanced labels. They choose N-gram features and run a feature selection process by choosing only the top 656 features ranked from Information Gain. Five base learners including Naive Bayes, SVM, Bayesian Network, C4.5 Decision Tree and Random Forest are selected to make the ensemble learner based on a majority voting method. The results show that their approach achieves F-measure as 84.2 and 91.7 for 2 classes and 3 classes tasks separately.

One highlight is a dimension reduction method Information Gain is applied to the data which help to reduce the size of the matrix. Also, the author considers the data imbalance problem which might potentially hurt the performance. By running a random resampling, a balanced dataset is built.

While one limitation is that they get their own data by using Twitter search API but didn't clarify the way they label the training dataset.

In [25], the author tested different combination of base learners, sentiment lexicons, and ensemble methods. They did the experiment based on the SemEval 2013 Task 2A dataset and tested Majority voting and Stacking methods on NB, SVM, SentiStrength lexicons. The overall results showed that the combination of three classifiers using majority voting achieved the highest score for both Tweets and SMS datasets with 86.05 and 88.82

respectively.

Compared with other works, they built the ensemble learners based on supervised learners and lexicon resource which is relative novel.

While they didn't describe how they train the model in a semi-supervised way which is part of their contribution. Also, only F-measure is provided in the evaluation part is quite limited.

In [48], the authors also evaluated the performance when using ensemble learning in Twitter data for sentiment analysis. They run the feature selection by using filter-based rankers (CS and ROC) with different parameters which consist 8 feature selectors overall. They also tried to run random under-sampling to deal with data imbalance problem. For ensemble learning, boosting and bagging were tested based on base learners like K Nearest Neighbors, C4.5 decision tree, Support Vector Machines, Multilayer Perceptron, and Logistic Regression. With a different combination of feature selection, data sampling, and ensemble learners' strategies, it is proved that using bagging and/or feature selection when training tweet sentiment classifiers are recommended for that they bring significant improvement. Also, it is good to use ROC when conducting feature selection on imbalanced Twitter data.

The highlight of this study is the authors considered the issues that might happen when dealing with real-world data. Few works discussed feature selection, data resampling, and ensemble learning by using different learners all in one study.

Interestingly, they found random under-sampling did make significant improvement compared with not using any technique at all, which is quite different from the conclusion drawn in [49].

The authors also have another related paper [33] which discussed combining feature selection with ensemble learning techniques to deal with the problem that high dimensionality present in Twitter sentiment analysis in addition to poor data quality or imbalanced data which has caused attention and could be solved by using ensemble learning. They discussed several kinds of feature selection methods including filter based, wrapper based, embedded or hybrid, and filter-based approach was the best to reduce computational resources. They proposed two techniques Select-Bagging and Select-Boost and seek to address both poor data quality and high dimensionality. The results showed that Select-Boost got the best performance compared with using no ensemble technique and also was significantly better than Select-Bagging for most classifiers on the 2 datasets.

The highlight of this study is they tried to solve the poor data quality and imbalanced data problem with one model.

While they tried to utilize both human-labeled and auto labeled tweets but the amount of the latter one (from sentiment 140) is quite small which might not be easy to update.

In [30], the authors built an ensemble classifier based on three individual classifiers, NRC, the GU-MLT-LT, and the KLUE, which were the top learners from SemEval 2013 contest other than using the classic classifiers like Logistic Regression or SVM. The key point of building this ensemble learner was to ignore the individual learners' classification decisions but acquire the classifiers' probabilities for each class. Then it chose the class with the highest average probability, and the final result was based on averaging confidences instead of voting schemes on the actual classifications of the individual classifiers.

One of the highlights is that they checked the influence of each individual classifier in the ensemble by running a Component Influence and an error analysis to analyze the misclassifications which showed that most of the errors observed concerns the misclassification of a tweet as neutral, which was not the worst misclassification possible.

Deep learning based approaches

Recent years, an increasing volume of studies start to focus on deep learning approaches to deal with sentiment analysis tasks. CNNs is one of the most popular models being used in recent years. Also, word embedding features are commonly used as a new feature for deep learning models.

In [31], a deep learning system was proposed for message-level Twitter sentiment analysis which ranked second on SemEval 2014 Task 9. Coooolll combines the sentiment-specific word embedding features and commonly used hand-crafted features. The sentiment-specific word embedding features which was based on C&W model (Collobert et al., 2011) tried to capture not just the syntactic features but also the sentiment information. The hand-crafted features included lexicon, Twitter-specific features, N-grams were used as well. The results showed that Coooolll outperformed other methods with an F-measure of 70.40 for 3 classes classification tasks.

The highlight of this study is their system was not hard to re-implement, especially for their unconstrained part which utilized emoticons as noisy labels without manually labeling.

While they didn't specify how to deal with the neutral part of data. Also, it would be better to show the running time for the unconstrained part which can give people a better idea of performance when re-implement.

In [32], a deep convolutional neural network was proposed to address sentiment analysis problems. The main idea of their approach was to utilize an unsupervised neural language model to train the base word embeddings model which could be further tuned with the help of a supervised corpus. At last, the pre-trained parameters could be used to initialize the model. Their model ranked top on Semeval-2015 Task 10.

The highlight of this study is they propose a new model to initialize the parameter weights of the CNN which helps to avoid injecting any additional features.

While it would be better to have another report based on bigger datasets to show the scalability of this model. So far, the sizes of training data being used in this research are pretty small.

An architecture was proposed based on CNNs (Convolutional Neural Networks) and LSTMs (Long Short Term Memory) in [34]. A big amount of unlabeled data was used to pre-train the word embeddings and a subset of unlabeled tweets were used to tune the model with distant supervision. After that, the CNNs and LSTMs were trained on the training dataset from SemEval-2017 and the embeddings got tuned again. The model ranked top for the five English subtasks with an F-measure of 0.748.

The highlight of this study is the combination usage of CNNs and LSTMs for sentiment analysis tasks. Also, the idea of the ensemble model did improve the performance further.

It would be better if training time could be shown as another metric especially, when an ensemble approach is being used.

Another similar study was done in [50]. To capture the contextual information from the corpus, word embedding features were trained in an unsupervised way based on GloVe.

Other than that, features like N-grams, Twitter-specific features, Word sentiment polarity score features based on lexicon were also used as part of the features. In the end, all the features were used as inputs of a CNN model which outperforms the baseline bag-of-words model and baseline SVM and Logistic Regression model using the same GloVe features.

The pro of this approach is that the experiment showed clearly how the use of different features can boost the performance (by comparing BOW and GloVe for SVM and LG) and how the use of CNNs outperformed SVM and LG (by comparing GloVe-SVM, GloVe-LG and GloVe-DCNN) in terms of different datasets.

It would be better if the author could analyze the computational complexity and also report the running time for the proposed approach. Alsp, there were no hardware usage details mentioned in this paper.

Instead of using GloVe as features, Word2Vec and Doc2Vec were tested in [51]. To do this, six datasets were built firstly based on STS dataset, dataset from [52], and dataset from [53] with different sizes for training and test purpose. After that, an ANN classifier was applied, and Word2Vec and Doc2Vec were tested respectively as features with different combination of training and test corpus. In terms of accuracy, 87.5 was achieved using Word2Vec (TN-w/e as training and STS-50K-w/oe as test) and 88.8 was achieved using Doc2Vec (STS-w/oe as training and STS-50K-w/oe as test).

The pro of this study is it tried to compare the performance for different word embedding features with different corpus size. It also showed the reasonable result achieved by only applying word embedding features.

It would be better if other performance metrics like precision, recall, and F-measure could also be reported for further comparison.

In [54], the authors tried to propose a system to overcome the problem that CNNs can extract higher level features and long-term dependencies but with the trade-off is a very deep neutral network with a big amount or layers. To make the model simpler and still efficient, a joint system was proposed in which RNN was firstly utilized for its ability to capture long-term dependencies; After that, CNNs was applied using global average pooling layer based on pre-trained word vectors using GloVe.

The highlight of this paper is a joint system was proposed to utilize the benefits from both RNNs and CNNs to reduce the complexity of the model with a reasonable level of performance.

It would be better if more performance metrics could be reported in the result. Currently, only accuracy was reported for the proposed method and the other compared works. Recall, precision, and F-measure were not used which makes it hard for a comprehensive comparison.

2.2.2.2 Semi-supervised learning based approaches

Supervised learning has achieved good performance for sentiment analysis, while it is not always available to get enough training data to train the model. It is not surprising to see that the works have been done by using a supervised learning approach are limited to use several well-known datasets which are time-consuming to build. However, it is much easier to acquire a big amount of unlabeled data from a social network with noisy labels. As a result, the semi-supervised learning approach becomes more and more popular for the ability to utilize as data as possible when high-quality labeled training data is limited.

The semi-supervised approach can be further divided into two categories: graph-based, self-training or co-training based. In the following section, we are going to review

how are they being used in sentiment analysis.

Self-learning & co-learning based approaches

Self-learning and co-learning are semi-supervised approaches which could utilize a small amount of high-qualify training data to build a model with by taking advantage of a much bigger volume of unlabeled or noisy labeled data. This approach is quite promising since it could extend the size of training data space significantly without human efforts.

In [57], the authors focused on the problem of imbalanced sentiment distribution in the social network leads to a bad performance on the minority class and the result of self-training is even worse compared with fully supervised method since the bias might be enlarged in each iteration. To solve this problem, they tried to improve the data selection strategy to enhance the performance of self-training on the minority class by using reserved self-training which employ selection on both labeled and unlabeled data. The result of the experiment on NLP&CC2012 dataset outperformed the best run by 2.06% macro-averaged and 2.30% micro-averaged F-measure.

The highlight is they proposed an enhanced self-learning model which was inspired by active learning is easy to understand and quite scalable. Also, the imbalanced problem was considered at the same time when training the ensemble classifiers.

It would be better if this approach could be tested on multi-classes tasks as well.

Another work about self-training is done in [58]. As a response for SemEval 2013, they tried to solve the subjectivity analysis or polarity detection tasks by using a single system. A subjectivity classifier and a sentiment classifier were built based on labeled Twitter training data which are combined into a hierarchical classifier. Additional unlabeled tweets were also used for the model to continue self-training. The system

achieved an accuracy of 61.2% using the hierarchical Naive Bayes model. The authors also mentioned that a lack of high-quality sources for additional objective-OR-neutral data was the biggest obstacle to increase performance compared with acquiring positive and negative tweets.

The highlight of this study is the system could be applicable in real life cases since the problem being discussed (a lack of high-quality neutral data) even though some result was not very desirable (the performance is quite low for negative class with an F-measure as 0.4786).

One concern is when processing the data, their proposed system would incrementally train one tweet at a time which might cause scalability problem when the volume of tweets is much bigger than now.

[59] also use self-learning method for SemEval 2014's task. In order to determine the given tweets are positive, negative or neutral, they chose several features including surface text, syntax, sentiment score, and twitter characteristic and built a classifier using SVM.

In the unconstrained setting by using self-training to add more unlabeled data in the training set, the performance decreased compared with the result of constrained setting. The author considered it as the low performance of the initial classifier. Also, a lack of training dataset could be another issue since the amount of training data for each domain was pretty small and a total of all the datasets only leads to 6k records. At last, it would be better if the author could clarify what corpus they used for self-training in unconstrained condition.

Apart from self-training, co-training is also being used in sentiment analysis tasks.

In [60], a framework was built by using co-training for Twitter sentiment analysis. The features were divided into text view features and non-text view features which are later contained into two separate random forest classifiers for training. Then the two classifiers were trained together collaboratively and periodically to improve the performance. When compared with SVM, classic random forest model, the proposed co-training model had a 3.8% improvement on the largest event set.

The usage of co-training was quite novel in Twitter sentiment analysis and also the authors tried to illustrate the sentiment orientation over time which might be helpful for building a real-time system in industries.

While the authors discussed to utilize the unlabeled data while co-training mainly solve the problem that the tweets are very short. Also, the paper was quite short, and some details are not discussed.

Graph based approaches

Graph-based approach is a series of models that try to take advantage of the graph structure to solve sentiment classification problems. Label propagation is one of the most commonly used one among the graph-based algorithms.

[61] discussed using label propagation, one of the most popular graph-based algorithms with its modification on sentiment analysis on datasets from multiple domains. The impact of different graph structure and parameter setting were tested in this study as well which provides a way to choose the most feasible algorithm based on available labeled and unlabeled data. Based on the experiment, the transfer learning approach GB-CDL was a competitive alternative to the fully supervised techniques and the semi-supervised method GB-SSL could help to obtain good results with a small amount of labeled data.

The highlight is the transfer learning approach could help to boost a semi-supervised model which shows the potential on similar tasks.

In another perspective, this method was domain-specific which might not achieve a good result like as this one if the test data is from another domain.

In [62], the authors aimed at solving the problem of high dimension in text analytics. Since traditional dimensionality reduction methods were usually unsupervised, they explored the availability of the sentiment label information to enhance the dimension reduction process. To do this, they proposed a Laplacian eigenmap (SS-LE) which remove redundant features by decreasing detection errors of sentiments and makes visualization of documents in low dimensional embedded space. In their work, their proposed method was compared with several dimension reduction methods which outperformed others in the experiment in terms of accuracy. It would be better if they could evaluation other measurements like precision, F-measure, recall as well.

One highlight is their work could be regarded as a good reference for dimension reduction research and decrease the computation cost since after processing, only a 2D or 3D space is being used as feature for the ML model.

Another work utilized graph-based semi-supervised approach for Twitter sentiment analysis is [12]. They tried to use social network relations by containing social relations and text similarities to build a graph-based semi-supervised classifier. The graph connected labeled and unlabeled points together which made the use of rich unlabeled data. The proposed SSA-ST model was tested on two datasets which got the highest accuracy of 86.2 and F-measure of 86.4.

In this study, features like social relations were used compared with the classic

syntax-based or semantic features. This kind of approach required powerful computational resources but could utilize more available features that other machine learning algorithms hard to achieve.

2.2.2.3 Unsupervised learning based approaches

Similar to semi-supervised approaches, unsupervised learning based methods also try to achieve accurate results without the usage of high-quality training data and manually labeling which could be expensive to acquire in some cases.

In [63], a fuzzy clustering method was proposed to deal with twitter sentiment analysis for a particular brand: Samsung Galaxy S6. The pipeline consisted of data collection, data preprocessing, unsupervised modeling and performance evaluation. With the tweets collected for one year of Samsung, three models were built which are K-means clustering, Expectation Maximization clustering, and the proposed fuzzy clustering method which combined PAM partition based algorithm as well as FANNY clustering algorithm. The result showed that their proposed model outperformed the K-means and EM based models.

The pro of this study is that it showed a way to not use any training data at all to address sentiment classification problems by applying comparing clustering methods.

While the first concern is there was no class level report on the performance, so we cannot know the performance of each class. In addition, even the proposed method which outperformed other two models suffer from a low recall (0.33), which means a lot of tweets with sentiment for that class were not detected by the model. Last but not least, this experiment was applied on a particular brand. It was unclear about the performance when apply it on tweets from another domain.

Since unsupervised approach cannot achieve accurate results individually,

sometimes it is used as a part of the system. In [64], the authors proposed a method for sentiment analysis on 6 categories of product reviews from Amazon. The major part of the model was to calculate the intensity of the sentiment words for each expression and a K-means clustering algorithm would be applied based on the intensity in the previous step.

The pro of this study is it tried to solve a real-time problem which can help customer to choose items online based on the sentiment polarity of reviews. Also, the imbalanced data problem was considered at the beginning which is universal in real-world data. In addition, the way of a combination of lexicon resource, key graph keyword extraction technique, and unsupervised K-means approach was quite novel.

2.2.3 Hybrid approaches

Although lexicon-based approach and machine learning-based (especially supervised ones) approach enjoy their popularity in Twitter sentiment analysis, there are still some remaining problems. For lexicon-based approach, sometimes there are no general opinion words in tweets since the words being used in Twitter is evolving over time; Also, the polarities could be domain-dependent so that one word which is positive in one domain could be negative in another domain. These problems make it hard for people to manually maintain a lexicon to satisfy the needs in Twitter sentiment analysis. While for machine learning approaches especially supervised learning approach, a lack of manual labeling training dataset for different domains is always a big concern. As a result, some proposed methods cannot be transplanted directly in terms of limited dataset and domains.

Since the existence of these problems, in recent years more and more researches start to turn to hybrid approach to combine lexicon-based approach and different kinds of machine learning approach together to utilize the advantages and compensate each other's

disadvantages of the above-mentioned methods to make the most potential of them. In these works, lexicon resources, machine learning algorithms, rule-based knowledge are combined together to solve the Twitter sentiment analysis problem.

In the paper of [65], the authors concluded the problem of individually using lexicon and machine learning approach and proposed a system to combine the lexicon-based and machine learning-based methods together. The system firstly utilized the lexicon-based approach to run sentiment analysis on Twitter data which leads to high precision but low recall. After that, a learning-based classifier was built by using extra training data to identify the polarities of the tweets from the result got by the lexicon-based approach in order to improve recall. The best performance the system was an accuracy of 85.4 and an F-measure of 74.9.

The highlight of this model is the authors made efforts to leverage the advantage of both lexicon and machine learning-based approach. A high precision and low recall meant the lexicon-based models were accurate but not flexible enough which is the pro of machine learning approach. In this way, they could offset each other's cons and it will not be surprising that a better result could be achieved.

One concern could be how to make sure that the performance of the lexicon part will not drop over time since the language style of tweets is changing.

[27] also proposed a hybrid system in dealing with Twitter sentiment analysis. As a response to SemEval-2013 Task 2, the system contained four components: normalization, rule-based classifier, lexicon-based classifier and machine learning classifier (SVM), in sequential order. When being called, the test tweets went through the pipeline in order and if a certain degree of confidence was not achieved, next classifier would be called. Each

step was not complex to run, so the system was quite applicable and not computational costly.

The upside of this pipeline is the results of each part was explainable and it could be a good system for business usage.

Some further improvements could be optimizing each classifier since the individual classifiers were pretty simple now and still had a lot of potentials. For instance, the rule-based parts and build of confidence threshold could be further improved.

In [28], the authors specified one common issue in Twitter sentiment analysis: there was no topic boundary in Twitter which makes the task harder than domain-specific tasks. As a response to SemEval 2013 task 2, they developed two models for constrained and unstrained conditions separately. The first one consisted of a lexicon-based model and a Naive Bayes tree model, while the second one contained a lexicon-based model and a maximum entropy classifier. For each of the model, the final results were fused by the two components at the posterior level.

It would be better to have more features test for the machine learning model apart from N-gram features since for each part, the components are not so complicated.

[66] is also a work using hybrid approach for Twitter sentiment analysis. They proposed a hierarchical model, starting by emoticon labeling if there is any emoticon appeared in the tweets, following by a step to label the unlabeled tweets in the first step based on a predefined list. For the tweets have not been labeled in the first two steps, a subjectivity lexicon was used to measure the overall sentiment polarity. Their approach was a linear combination of using noisy labels and lexicon which achieved 73.72 accuracy.

One problem of this approach is that in the first step, emoticons were used as final

labels which could introduce some noise since they are usually be regarded as noisy labels; Also, the predefined list was a limited list of strong sentimental words which could hardly exhaust the words in twitter that have strong emotions; Furthermore, the third step was based on SentiWordNet which means it might be a lack of flexibility since all the parts were all static method which won't evolve with tweets over time.

In [67], they defined the primary issue for Twitter sentiment analysis was low accuracy and data sparsity, also a big number of tweets classified as neutral. To solve the above-mentioned problems, they proposed a real-time system which contains three modules including a data acquisition step to obtain tweets feeds, a pre-processing and transforms step to perform different transformations, and a last step to do the classification work. The classification part contained three classifiers: emoticon classifier, N-gram based machine learning classifier, and SentiWordNet driven lexicon classifier in order. The best accuracy was achieved on dataset 1 with 88.89%.

The upside is the authors brought these issues into consideration which were quite universal in Twitter sentiment analysis tasks.

While one concern of this work w that the number of tweets collected from Twitter API was quite small (the biggest dataset only contains 1000 tweets). Also, it would be better if they can go into details about the real-time architecture.

Focusing on Twitter sentiment analysis in a business environment, the authors in [68] identified some common problems in Twitter sentiment analysis: the difficulty of data gathering, the universal existence of neutral tweets that mentioned brands, existing approaches suffering from low accuracy, and a limited amount of test dataset.

Especially, they did a thorough analysis in feature engineering part and discuss why

barely using TF-IDF, which was one of the most commonly used feature representation methods, cannot capture all the useful information in tweets. They also proposed a novel feature engineering method for Twitter sentiment analysis tasks. Another highlight of this paper was it arise attentions to the universal problems of Twitter sentiment analysis in brand management which made it closely related to a real business case. Also, they put efforts to build a Twitter-specific lexicon for sentiment analysis tasks.

[69] introduced several toolkits for aspect parsing and Common-Sense Knowledge-based Sentiment Analysis which works as a sequential workflow. For the sentiment analysis part, a combination of rule-based method and ensemble learning-based approach was proposed. For every sentence in the test dataset, if there was at least one concept could be found in SenticNet, the knowledge-based method would be used to detect sentiment polarity. Otherwise, the machine learning-based approach would be employed.

The upside of this study is they showcased some individual sentence as an example when analyzing which could be helpful to understand how the rules work.

For their work, it would be better to have a more detailed evaluation. Also, no sentiment analysis results on an aspect level were present.

In [70], the authors utilized Twitter sentiment analysis to get a better understanding of the political opinions of the users. They proposed a framework from data collection, data storage, data preprocessing, sentiment classification, and sentiment orientation validation to the performance of different sentiment lexicons. Basically, the authors got 1690 tweets after data preprocessing and run the sentiment classification task based on three methods: SentiWordNet, W-WSD, and TextBlob; In order to validate the result gotten from the above three methods, two machine learning models, Navie Bayes and SVM, were used as

comparisons. As a result, SentiWordNet achieved the worst result compared with W-WSD and TextBlob when applying Naive Bayes and SVM; In contrast, W-WSD worked better in Naïve Bayes based model, and TextBlob worked slightly better in SVM based model.

The pro of this research is it provided a complete framework for Twitter sentiment analysis from the start to the end. The way they utilized hashtags to crawl tweets could be duplicated easily in other researches. Also, they focused on a specific domain democratic parties in India which could be helpful to other fine-grained researches;

One of the cons is there was no process to build a gold standard of the training dataset. The authors collected tweets via Twitter's API, but no matter what methods they used, lexicon, or machine learning-based, the accuracy of the results were not guaranteed. Therefore, there is some concern about using machine learning approach to validate the result of different lexicon approaches. Also, it would be better if they can clarify the impact of the auto-translation on getting a non-biased sentiment orientation result.

In [71], the authors aimed at sentiment analysis on Arabic Tweets. In this study, different combinations of semantic features, stylistic features, and Twitter specific features were tested to get the best result. Also, they chose backward selection from best-subset selection, forward stepwise selection and backward stepwise selection as the feature selection method since they got better performance compared with Information Gain and Chi-Square. Their proposed classifier was a hybrid one with the knowledge from lexicon and machine learning method (SVM). In the experiment, the collected data were annotated as four classes: positive, negative, neutral, and mixed. Correspondingly, two-way (positive and negative), three-way (positive, negative, and neutral), and four-way (positive, negative,

neutral, and mixed) classification were tested with the best F-measure of 69.8 (NE), 60.71 (EM), and 53.56 (EM) for the three tasks collectively.

The pro of this paper is its focus on the Arabic tweets which is more challenging compared with English tweets (the author mentioned there was Twitter-specific POS tagger like the CMU Twitter NLP tool for English but nothing like that for Arabic tweets). Also, they proposed the backward selection outperform other traditional feature selection method which is not commonly used before. They also tested a bigger size of corpus than the previous works did.

It would be better if they could show not only the F-measure but other metrics like precision, recall, and accuracy. Also, the amount of mixed class tweets was significantly lower than the other classes, and the imbalanced class problem could be handled better in future works.

[72] is a study to extract sentiment orientation for Oman tourism from tourists via Twitter. They proposed a hybrid approach which was to test four kinds of features: domain specific ontology features, entity features, lexicon features, conceptual feature. For domain-specific ontology features, they built their own ontology with nodes as concepts and edges as the relationship between concepts; Specific lexicon resources were also built based on three existing lexicons such as Sentistrength, SentiWordNet and Opinion lexicon; Conceptual semantic features which contain outside knowledge were used when building Navie Bayes classifier. In the following experiment, the effectiveness of these four features were tested and the one using conceptual semantic features achieved the highest F-measure (85.54).

The pro of this paper is the consideration of conceptual semantic features which

could help in the cases that no sentiment words were explicitly used in the tweets; Also, building domain-specific ontology was a novel way to deal with sentence-level sentiment analysis;

While one of the cons is the details of how to apply the four types of features were not thoroughly explained which make it unclear for people to duplicate this study; Also, the process of data collection and annotation were not mentioned at all, and there is no way to validate the accuracy of the labels of training dataset.

In [73], a hybrid approach was proposed to deal with streaming Twitter sentiment analysis. A framework including data collection via Twitter API, data preprocessing, feature generation, hybridization modeling, and streaming data fit in were introduced.

A highlight is a hybrid approach which was a linear combination of particle swarm optimization (PSO), genetic algorithm (GA), decision tree (DT) were used as the major modeling part which achieves an 86.9 F-measure.

It is a pity that the authors didn't explain the hybrid method in detail, like, how the three major components worked with each other. Also, there was no proper annotation for the hybrid modeling algorithm. Another con is it was claimed that 600 million tweets were collected and used in this study, but there was no explanation of how these tweets are annotated as the training dataset.

An interesting approach is proposed in [74]. Compared with other studies, the authors focused on the effectiveness of sentiment diffusion patterns on Twitter sentiment analysis which was rarely used in previous works. By utilizing Repost Cascade Tree (repost relationship among tweets) and Repost Diffusion Network (repost relationship between users), Sentiment Reversal phenomenon was measured, and several features were extracted

based on these two graph structures which could help to build a sentiment reversal model based on SVM. The proposed SentiDiff algorithm combined the sentiment reversal model and textual information-based model with the assumption of if the prediction results between the two models were conflicting, the textual information-based model had a higher chance to make incorrect predictions. In the experiment, sentiment reversal prediction achieved an 80.91 F-measure when applying all features and the SentiDiff algorithm leaded to a 5.09% to 8.38% increase based on some existing models.

This study has several pros: Firstly, the process of tweets collection and annotation were described in detail and processed under an unbiased standard which ensured the data quality in the training dataset; Moreover, the imbalanced data problem was also considered and dealt with based on the method in [74]; The highlight is they firstly took diffusion features into account which contribute to a better performance.

While on of the concerns could be: The improvement had to base on the completeness of the repost cascade trees and repost diffusion network structures which could be hard to acquire in real life, computationally intensive, and space consuming to be stored. Also, it would be better if they can also show the F-measure for the SentiDiff algorithm for comparison with other studies.

An interesting study was done in [26] to employ the topic model to enhance Twitter sentiment analysis. They noticed the problem of the previous models which use a variety of features that only utilize local information whose meaning might vary over different topics. In order to train a model which might capture the different meaning of words in different scenarios, they proposed an approach with a self-training process to utilize millions of unlabeled tweets as well to enrich the corpus and run an LDA model to divide

the tweets into different topics and train models for each topic. When testing on SemEval 2013 task B dataset, a 69.7% F score was achieved.

The upside of this approach is that it contained an iterated self-training process which could utilize a huge number of unlabeled tweets to facilitate the model training process since compared with the training data, the cost of acquiring these unlabeled tweets is relatively low.

While there were also researchers run a similar test based on their method and suffer from a lower performance. They proposed that for some topics, there was not enough training data for some values since the default topic number is set to 100.

In [55], the authors supposed that the popular unsupervised learning approach like LDA requires providing the specific number of topics which will introduce some subjectivity. Also, the model tended to treat fact words and opinion words the same and assume that for one tweet there was only one topic used. In order to address edthese problems, a hybrid Hierarchical Dirichlet Process-Latent Dirichlet Allocation (HDP-LDA) model was proposed in this study to solve the above-mentioned problems.

The strength of this study is it focuses on the limitation when using the traditional LDA model when dealing with sentiment analysis tasks which might have a more general usage when addressing other text mining tasks. Another highlight is it could help to generate the opinionated positive and words for in aspect based which could be helpful for online customer review.

While some concerns could be that in the paper the authors were using perplexity for parameters setting and comparison with other models, but perplexity is considered as a bad measurement for topic models which only consider the individual words in the topics

but not the context and the entire coherence within each topic. Therefore, the usage of perplexity tends to decrease the credibility of the research. This method could be inspiring for people that want to utilize topic models but maybe not for tweets since the assumption in this paper (one sentence doesn't contain only one topic) and different language style and evolving speed make it might not be applicable for tweet sentiment analysis.

In [56], the authors found that most of the commercial tools for sentiment analysis were limited to distinguish the sentimental orientation for the whole document but not in an aspect level. Also, they failed to capture the implicit meaning but only the explicit sentiments which makes them easy to be tricked by negation and disjunction. In their study, a model called Sentic LDA is proposed to integrate common-sense reasoning with LDA algorithm for a deeper analysis from syntax to semantics aspect-level sentiment analysis.

The strength of this study is they enhance the existing LDA model by combining common-sense computing into it to boost the performance. This was achieved by indicating μ as the contribution of common-sense knowledge for determining β, which was the parameter used to determine the topic distribution in a classic LDA model.

One downside might be several parts of this model require manually labeling which makes this model less automated. For example, the authors mentioned that "clusters generated by LDA are manually labeled by aspect category" and "we manually labeled each aspect term by its corresponding aspect category".

2.2.4 Literature summary

In Table 2, a comparative table is shown as a summary of the 48 papers we reviewed for sentiment analysis.

From the 48 papers we reviewed, we can see that Twitter sentiment analysis has drawn a lot of attention these years. Generally, lexicon-based approach is one of the first methods that to be used for sentiment analysis tasks at an earlier stage but is not usually used for Twitter sentiment analysis individually due to the trait of tweets is different from other types of text; Instead, machine learning approach is one of the most popular approaches used for Twitter sentiment analysis with a variety of types and algorithms. Specifically, supervised learning approach enjoy its popularity since the beginning till now by starting with a single classifier to ensemble learning and deep learning methods; Meanwhile, semi-supervised and unsupervised learning are also drawing more and more attention to offset the lack of high-quality training data in some cases. While recently, hybrid methods play a leading role for a number in Twitter sentiment analysis task by combining the pros of different existing approach and also compensate the cons for each other too. It's not rarely seen that a combination of lexicon, graph-based method, supervised learning and deep learning in the most recent studies. This trend means the maturity of this field since people are getting a better understanding over time about the benefits and shortcomings for different individual approach. Works about sentiment analysis on Twitter emerges with the increasing influence of Twitter as a social network all over the world. Although the lexicon-based approach has a long history, while studies of Twitter sentiment analysis didn't start with lexicon-based approach since the

performance of using lexicon on Twitter is pretty poor because of the use of informal language, short text length, no domain boundaries and abundance of noisy symbols.

As a result, it is better to use Twitter's own data as training data to feed into the model to deal with sentiment analysis on Twitter. Therefore, the supervised learning approach is firstly tested for Twitter sentiment analysis tasks. As for models, Decision Tree, Naive Bayes, SVM, Logistic Regression are the most popular supervised learning. When it comes to features, N-gram, POS, negations which are the also used in sentiment analysis for other domains are also tested in different researches. Then we can see the way to build the model, and the features being used are both evolving. The ensemble learning approach is introduced into Twitter sentiment analysis which could utilize several weak learners to build a strong classifier; Also, a variety of Twitter-specific features are gradually added in studies about Twitter sentiment analysis.

The potential of the supervised learning approach is limited by the availability of high-quality training data, especially when it comes to Twitter. This inspired the researchers to explore semi-supervised/unsupervised and hybrid approaches for TSA to make the most use of accessible training data and integrate the big volume of unlabeled data especially when there is a plenty of posts out there accessible in social network which will keep being produced by users. Also, the results are not satisfying by directly using lexicons into Twitter sentiment analysis. There is a trend to utilize the knowledge of lexicons to boost the potential of the existing machine model. We can see that the semi-supervised approach like self-training and co-training are tested with an integration of training data and lexicon resources which achieves good performance with limited available resources. What's more, hybrid approach with different combination of different lexicons, supervised

Table 2 Literature Summary of Sentiment Analysis Approaches

type	subtype	paper	year	Algorithm/Model	lexicon used	dataset	domain
lexicon	-	[3]	2011	-	Null	Epinions 1, Epinions 2, Polarity Dataset[75], Camera Dataset	-
lexicon	-	[29]	2015	-	Bing Liu Lexicon, MPQA Subjective Lexicon and SentiWordNet	SemEval 2014 Task 4 dataset	Twitter
lexicon	-	[41]	2018	-	SentiWordNet	self-collected	Twitter
supervised	singer classifier	[43]	2010	NB	Null	self-collected tweets from accounts of 44 newspapers (for objective posts), tweets with emoticons (for positive and negative tweets)	Twitter
supervised	singer classifier	[44]	2011	SVM	Dictionary of Affect in Language (DAL), WordNet	annotated Twitter data (tweets) from a commercial source	Twitter
supervised	singer classifier	[23]	2013	SVM	NRC Emotion Lexicon, MPQA Subjectivity Lexicon, Bing Liu's Lexicon	SemEval-2013 dataset	Twitter
supervised	singer classifier	[24]	2014	SVM	WordNet 3.0, NRC Emotion Lexicon, Bing Liu's Lexicon, and the MPQA Subjectivity Lexicon, Roget's Thesaurus, Sentiment140 Lexicon	SemEval-2013 dataset	Twitter
supervised	singer classifier	[6]	2014	Perceptron/Ensemble Technique	Null	Sanders Corpus Data Set	Twitter
supervised	singer classifier	[45]	2019	NB, SVM, RF, CNN, LSTM, FCDNN	Null	Obama-Mccain Debate(OMD), Sentiment140	Twitter
supervised	ensemble learning	[7]	2014	RF, SVM, NB, LG	Ppinion lexicon3 proposed by Hu and Liu	Sanders — Twitter Sentiment Corpus, Obama-McCain Debate (OMD), Health care reform (HCR)	Twitter
supervised	ensemble learning	[47]	2015	NB, SVM, BN, C4.5, RF	Null	self-collected	Twitter

supervised	ensemble learning	[25]	2015	NB, SVM, SentiStrength lexicon	Bing Liu Lexicon, MPQA Subjective Lexicon, AFINN Lexicon (AFINN)	SemEval-2013 dataset task 2 A	Twitter and SMS
supervised	ensemble learning	[48]	2015	KNN, C4.5, SVM, MLP, and LG	Null	sentiment140 corpus	Twitter
supervised	ensemble learning	[33]	2015	KNN, C4.5, MLP and LR	Null	2015 SemEval Task 10, subtask B dataset, sentiment140 corpus	Twitter
supervised	ensemble learning	[46]	2015	KNN, two versions of C4.5, SVM, MLP, RBF, LR	Null	sentiment140 corpus	Twitter
supervised	ensemble learning	[30]	2015	classifiers from NRC-Canada, GU-MLT-LT, KLUE	Null	SemEval 2013's task 2 and SemEval 2014's task 9	Twitter
supervised	ensemble learning	[11]	2018	NB, RF, SVM, LR	Null	Stanford - Sentiment140 corpus, Health Care Reform (HCR) ,First GOP debate twitter sentiment dataset, Twitter sentiment analysis dataset (from kaggle)	Twitter
supervised	deep learning	[31]	2014	word embedding	4HL, MPQA Subjective Lexicon, NRC Emotion, NRC Hashtag Sentiment Lexicon, and Sentiment140 Lexicon	SemEval 2014	Twitter
supervised	deep learning	[32]	2015	CNNs	Null	SemEval 2015 task A (phrase level) and B (message level)	Twitter
supervised	deep learning	[34]	2017	CNNs and LSTMs	Null	SemEval 2017	Twitter
supervised	deep learning	[50]	2018	GloVe-DCNN	AFINN	self-crawled, STS, SemEval2014 Task9, STSGd, SED, SS	Twitter
supervised	deep learning	[51]	2018	ANN	Null	STS, dataset from [52], dataset from [53]	Twitter
supervised	deep learning	[54]	2019	CNN, RNN	Null	self-collected data, STS, SS, HCR	Twitter
unsupervised	-	[63]	2016	Fuzzy Clustering	Samsung Galaxy S6	Mobile phone	Twitter

unsupervised	-	[64]	2017	Kmeans	MPQA Subjective Lexicon	product reviews (camera, laptop, mobile phone, tablets, TVs and video surveillance devices)	Product reviews from Amazon
semi-supervised	graph-based	[61]	2013	Label Propagation	Null	Amazon product reviews on 4 topics: books(BO), electronics (EL), kitchen appliances (KI) and DVDs (DV) (Blitzer et al., 2007)	mutiple
semi-supervised	graph-based	[62]	2014	SVM, k-NN	SentiWordNet 3.0	Book, DVD, Electronics, and Kitchen reviews	Product reviews from Amazon
semi-supervised	graph-based	[12]	2015	SSA-ST	Null	Health Care Reform(HCR), Obama-Mccain Debate(OMD)	Twitter
semi-supervised	self-learning & co-learning	[57]	2013	MaxEnt and SVM	HowNet, NTU Sentiment Dictionary, WI sentiment analysis lexicon	NLP&CC2012	Tencent Microblog
semi-supervised	self-learning & co-learning	[58]	2013	Multinomial Naive Bayes	Null	approximately one million tweets having emoticons, In addition to collecting unlabeled data using emoticon keywords, sentences from Wikipedia as neutrally labeled text	Twitter
semi-supervised	self-learning & co-learning	[60]	2013	RF	WordNet	4 Twitter datasets (US Unemployment, American Train Service, BBC World Service Staff Cuts, Obama Election)	Twitter
semi-supervised	self-learning & co-learning	[59]	2014	SVM, LG	MPQA, SentiWordNet, NRC, IMDB	LiveJournal(2014), SMS2013, Twitter2013, Twitter2014, Twitter2014Sarcasm	Multiple
hybrid	topic model	[55]	2013	HDP-LDA	SentiWordNet	restaurant review collected in [76]	Restaurant reviews

hybrid	topic model	[26]	2014	LDA, Smoothing technique	lexicon from Bing Liu, NRC Hashtag Sentiment Lexicon	task B of Sentiment Analysis in Twitter in SemEval- 2013	Twitter
semi-supervised	topic model	[56]	2016	enhanced LDA	SenticNet and extension	Semeval 2014 restaurant dataset	Restaurant reviews
hybrid	-	[65]	2011	-	Ding et al., 2008 lexicon	Obama; Harry Potter; Tangled; iPad; Packers	Twitter
hybrid	-	[27]	2013	SVM	SentiStrength	SemEval 2013 task 2, subtask B	Twitter
hybrid	-	[28]	2013	NB	Based on the one presented in [77], which in turn is an expansion of [78].	SemEval 2013 task 2, subtask B	Twitter
hybrid	-	[66]	2013	-	SentiWordNet	Stanford University consisting of 60,000 tweets	Twitter
hybrid	-	[68]	2013	DAN2, SVM	Twitter-specific lexicon set	10345184 tweets from May 6th at 7am to June 8th at 7am	Tiwtter
hybrid	-	[67]	2014	-	Null	Twitter dataset about Imran Khan, Nawaz Sharif, Dhoni 100, Tom Cruise, Pakistan, America	Twitter
hybrid	-	[69]	2014	-	Wordnet synonyms, SenticNet 3.0	Blitzer dataset	-
hybrid	-	[70]	2018	NB, SVM	SentiWordNet, W-WSD and TextBlob	self-collected	Twitter
hybrid	-	[71]	2018	SVM	AraSenTi lexicon, the Arabic translations of the lexicons in: MPQA and Bing Liu.	self-collected	Twitter
hybrid	-	[72]	2019	NB	Sentistrength, SentiWordNet and Opinion lexicon	self-collected	Twitter
hybrid	-	[73]	2019	particle swarm optimization (PSO), genetic algorithm (GA), decision tree (DT)	Null	self-collected	Twitter
hybrid	-	[74]	2019	SVM	Null	self-collected	Twitter

learning, semi-supervised approaches also enjoy its popularity in recent years. Apart from the overview mentioned above, there is also some trend that is worth our attention:

- Finer-grained improvement is witnessed as an increasing trend.

The most recent studies are paying more attention to improve each component of the sentiment analysis system like the building of lexicon resources, the labeling of datasets, the representativeness of features, creating new ways for feature selection, etc. For example, a new feature representation way (Quantum-inspired Sentiment Representation) is proposed in [45]; A new feature selection (backward selection) is proposed in [71]; New features like phrases base features is proposed in [45], and Cascade Tree Feature and Diffusion Network Feature are proposed in [74]; A novel slang dictionary is built in [41].

- Researchers tend to build their own dataset/lexicon/resources in recent studies.

When comparing the dataset used for Twitter sentiment analysis, a trend is witnessed that more and more self-built corpus is used in recent studies than before to meet the needs of the research. The first reason is the changing policy of Twitter like we mentioned in the previous section that restrict the distribution of tweets, which makes few publicly available datasets are available. Furthermore, with the popularity of deep learning approach being applied to this field, billions of tweets are needed to pre-train the word-embedding features or to train the model. The traditional human-labeled datasets are no longer sufficient for these kinds of models. Therefore, it is not surprising that more people are self-collecting the datasets they need. While the side-effect of this is it makes the comparison between approaches becomes much harder.

- Language specific Twitter sentiment analysis is drawing more attention.

It is clear that most of the Twitter sentiment analysis studies are focusing on English tweets. However, the reality is Twitter is a mixture of usage of different languages where

no language limitation is applied for users when posting. So that instead of only focusing on English tweets, more and more studies start to discuss the achievements and challenges facing in other languages. For example, In [57] the problem the authors try to solve is sentiment analysis on Chinese microblogs; In [71] the authors focus on the difficulty of sentiment analysis on Arabic tweets; Moreover, SemEval-2017 provides a set of human-labeled Arabic tweets for research usages which is a response of the increasing attention in Arabic related studies.

2.2.5 Gap analysis for sentiment analysis approaches

Specially, we want to clarify the problem we identified based on the previous studies we reviewed.

Firstly, for the works that were using training datasets that built by noisy labels, there is a lack of semantic topic detection, which will lead to decreased performance of models due to synonymy and polysemy issues. Detecting semantic topic behind tweets is essential when using training dataset that is built by noisy labels since there is no topic limitation in Twitter compared with other platforms. For the tasks like sentiment analysis of product reviews or restaurant reviews, there are topic limitations naturally: The customers tend to focus on the discussion about the product they have bought or going to buy, or the food, environment, service, location of the restaurant they visited. In contrast, the nature of Twitter makes it a platform that people can freely post or discuss any topic they would like to without any topic limitation, which will bring some challenges to the researchers. Without topic limitation, one potential impact is the synonymy (several different words share similar meanings) and polysemy (one word could have several different meanings or part of speech) issue. For any sentiment classification model that is based on the bag-of-word model, these two issues, especially polysemy, will bring noises

to the model since one word is usually considered as one point in the bag-of-word model. For Twitter sentiment analysis, a lack of topic limitation of training dataset will emphasis the synonymy and polysemy issues compared with the tasks having topic scopes.

Secondly, there is no automated way to find out the best topic model by choosing the most proper topic numbers for Twitter sentiment analysis. Topic modeling is the task to uncover the latent topics behind a collection of tweets, which can be regarded as a potential approach to eliminate the no topic limitation challenge for Twitter sentiment analysis. While for some popular topic models (the actual models used to achieve the goal of topic modeling) like Latent Dirichlet Allocation (LDA), one concern is that the number of topics is a required parameter when building the model which is hard to choose without having a deep understanding of the dataset. One previous work [26] set the topic number to 100 as a fixed number when building the model, while another work [55] choose to utilize Hierarchical Dirichlet Process (HDP) model to automatically decide the topic number. While none of these methods can guarantee that the topic number, they choose can facilitate to build the best model on topic of the current training dataset. Therefore, there is a need to propose a method that can find out the best topic model by choosing the best topic numbers.

2.3 Twitter Sentiment analysis use case studies and online tools

2.3.1 Use case studies and online tools review

In this section, we will focus on reviewing the existing use case studies and online tools for Twitter sentiment analysis or general sentiment analysis.

A topic model-based sentiment visualization system was proposed in [79]. The overall system consisted of three major components: topic detection, sentiment analysis, and visualization. LDA was utilized for topic modeling in this research and symmetric KL-

Divergence was applied to find the best number of topics. In order to improve the performance of topic modeling, stop words were removed from the data, and only nouns, verbs, adjectives, and numerals were kept fitting into the LDA model. For sentiment classification, SentiStrength and SentiWordNet were used as lexicons for sentiment classification part.

The highlight of this study is the interactive graphic interface was designed to correct the wrongly classified cases from the result of the lexicons. The users could review the sentiment polarities for different aspects and correct the wrong ones.

One concern is the system is quite static without parts of data collection and data preprocessing. Also, human efforts were needed every time a new analysis is run, which makes it less flexible for different inputs.

An interesting study is run in [80]. A web system named iFeel was proposed to detect sentiment polarity on top of seven existing tools. User could input a single sentence or upload a document with no more than 10,000 lines of messages. After getting the input, sentiment classification would be applied on all seven tools, including SentiWordNet, Emoticons, PANAS-t, SASA, Happiness Index, SenticNet, and SentiStrength. The user could assign different weights on these tools to make it fit for their needs, like, increase the weight for SASA for political content related tasks to improve the performance.

There is another following study of [80] that is published two years later. In [81], iFeel 2.0 was proposed with the integration of 19 sentence-level sentiment analysis methods. In addition to the methods that were used in iFeel, iFeel 2.0 also added Opinion Lexicon (Hu and Liu 2004), Opinion Finder (Wilson, Wiebe, and Hoffmann 2005), AFINN (Nielsen 2011), SO-CAL (Taboada et al.), Emoticons Distant Supervision(Hannak et al. 2012), NRC Hashtag (Mohammad 2012), EmoLex (Mohammad and Turney 2013), SANN

(Pappas and Popescu-Belis 2013), Sentiment140 Lexicon (Mohammad, Kiritchenko, and Zhu 2013), Stanford Recursive Deep Model (Socher et al. 2013), Umigon (Levallois 2013), Vader (Hutto and Gilbert 2014). All those methods were deeply discussed in (Ribeiro et al. 2015). The architecture was also updated to Meteor, and there was better support for Multilanguage sentiment analysis.

The pro is the author validate that integrating several different tools can help to improve the F-measure which was like an ensemble learning approach. They also evaluated the scalability of the system by measuring the CPU time required when running with different size of data.

One concern is that there was a missing part of validating that adjusting the weight can help to improve the domain-specific classification performance since the new weight would be applied on all the data which might convert some correctly classified cases into the opposite polarity. Therefore, it was unclear if changing the weight will help to improve the performance, or, how much it would improve.

In [82], a Social Sentiment Sensor (SSS) system is proposed based on Microblog in China to detect the trending topics and public sentiment towards them. There were two major components of the system: Hot topic detection and Topic-oriented sentiment analysis. Hot topic detection aimed at finding the most discussed topics for the users during a period of time which consists of Topic detection, Topic clustering and Topic popularity ranking; Topic-oriented sentiment analysis utilized the results from the previous step to determine the sentiment polarity in a topic based level by collecting topic-related messages, detecting message sentiment, and summarizing topic sentiment distribution in terms of five emotions: happy, sad, angry, surprise, and fear based on 3357 manually labeled training data. The dashboard showed the ranks of the topics and the corresponding sentiment scores

towards them, as well as a trend over time and a location breakdown.

The pro of the study is it had an architecture from topic detection to dashboard illustration, which also considered the schedules to run the model periodically and data freshness checks.

One of the cons is there was no “fear” related labeled data in the training dataset as described which makes it unclear how the classifier could work and achieve a result with “fear” in it. Also, it would be better for the authors to have another section to introduce how they deal with streaming data since they mention the dashboard is refreshed on an hourly basis. In addition, the training dataset was manually labeled which could lead to some automated concerns of the system over time.

In [83], a system was proposed to mine data in Chinese microblogs which facilitated data collection, topic extraction, and sentiment classification. Their data collection section was driven by Kafka, a distributed messaging service, to receive the real-time data in an incremental way, and processed by Storm, a distributed real-time computation system, for Data Preprocess and Persistence, and Realtime Data Statistics. The data collection step was followed by topic analysis and sentiment analysis in a parallel way. The topic analysis was applied to uncover the latent topic and for hot degree topic and topic tracking purpose. This part was implemented on top of a 4-node Hadoop cluster. In addition, the sentiment classification was driven by a hybrid lexicon-based approach which was the integration of Emotional symbol dictionary, Popular phrasing dictionary, and NTUSD sentiment dictionary. For the visualization part, sentiment scores in terms of provinces were shown. Topic evolving tracking was also supported in this tool.

The pro of this study is an entire architecture was introduced from data collection to data visualization based on Kafka, Storm, and Hadoop which was scalable with the

increase of data volume. Also, it highlighted the topic visualization part by ranking the most popular topics.

One concern is since this study was an extension of their previous work, especially for the sentiment analysis part, there was no evaluation in this work about the performance of their proposed lexicon-based hybrid classifier.

In [84], the authors focused on using Twitter sentiment analysis on public opinion towards Malaysia property industry to uncover the imbalances in supply and demand. A pipeline was built containing data extraction, data preprocessing, sentiment classification, and data visualization. For data extraction part, 745 tweets with keyword equal to PR1MA or #PR1MA were collected which are about government affordable housing project. The sentiment classification was applied based Naïve Bayes algorithm in R based on the manually labeled training data. Tableau was used as the visualization tool to showcase the overall volume, sentiment distribution, and real-time trend.

The pro of this paper is the effort made to help to solve a real problem based on Twitter sentiment analysis which helped to uncover the opinions around the PR1MA.

One of the cons is the training dataset only contained 420 data which might be a potential issue for long term streaming data analysis since the language style in Twitter tend to change over time. Also, there was no description of how the authors ran the aspect level sentiment analysis that shown in the dashboard. There is also a missing part in the architecture section about how to receive streaming tweets via Twitter Streaming API.

An interesting study was done in [85] which focus on Twitter sentiment analysis in terms of location data. The proposed system run by starting with asking the user to provide a keyword (could be event name or a hashtag). After that, the request would be intercepted based on Flask framework and the data collection process will be triggered by utilizing

Twitter API. Once the data was ready from the previous step, data preprocessing and sentiment classification including subjectivity analysis and polarity analysis, would be applied, which returns the subjectivity and sentiment scores. In addition, the output of the previous step would be used as input of data visualization to show the subjectivity and polarity results based on location. The whole process was described based on the case study US Presidential elections of 2016.

The pro of this study is the authors focused on a geo-based approach to analyze Twitter sentiments which was helpful for political events like US Presidential election. The table in the end which compared the eventual winter for each state is quite close to the polarity score calculated based on the proposed system except Illinois and New York. Also, they created a geo data dictionary for US by improving the portion of mapping based on user-provided locations.

The con of this research is the sentiment classification was applied based on Python TextBolb which was a lexicon-based approach. Politically related tweets were one of the toughest tasks in Twitter sentiment analysis since the wording could contain complex meaning that hard for classifiers to capture. In the paper, the author did not validate the performance of using TextBolb on the collected tweets.

[86] is a study that focused on sentiment analysis of Chinese tourism blogs and reviews. The goal of this system was to help customers to understand the up and downside of different tourist attractions. The blogs and reviews were from four tourism websites: Ctrip, Mafengwo, TripAdvisor, and Tuniu and the data related to Jeju Island was used in this study. After data collection, THULAC (THU Lexical Analyzer for Chinese) was used for data preprocessing due to its good performance. In addition, eight properties were manually defined for fine-grain breakdown later including Price, Landscape and

Attractions, Mood, Entertainment (Interest), Experience, Politeness and Kindness, Regional Features, Accessibility. For each property, a property dictionary was created by searching for the synonyms and related words from search engine and later on, the data would be categorized into different categories. Random forest was used for the sentiment classification part. For the data visualization part, firstly a time-series graph was used to show the trend of sentiment score over time; Also, a radial graph was used to show the scores for different properties; Furthermore, a network graph was used to show the relations among different words with positive and negative sentiments.

One of the pros of this paper is its visualization brought fruitful information for the user to take a quick glance of a tourism attraction within a short time. Also, the idea of breaking down the properties facilitated a finer grain analysis compared with only have an overall score.

However, one of the cons is the way the documents being categorized based on the property dictionary was by comparing whether a word exists in the dictionary or not, which means one document could be categorized into different property categories and there was no validation about the accuracy of this approach; Also, the authors evaluated the user experience of their tool by measuring the time reduction of the users before and after using the tool. While, a better way was comparing the time reduction by using their tool and other tools.

A system was proposed in [87] for sentiment analysis based on data from BBC news. Firstly, a pronoun replacement-based text summarization was used to replace all the pronouns appear to the corresponding nouns, which was a part of the preprocessing; Furthermore, VADER, a rule-based sentiment analyzer was used for sentiment analysis.

The visualization of sentiment analysis results was shown in a 3D graph for 20 news articles.

The highlight of this study is the use of pronoun replacement-based text summarization which would replace pronouns with proper nouns; Also, the authors tried to use a new way to visualize the sentiment analysis results with several dimensions.

Although the 3D representation idea was novel, the graph itself was not a very clear way to convey the information and sometimes make it hard for people to read.

In [88], the authors proposed a system for Twitter sentiment analysis and visualization since Twitter was still a quite open public opinion source compared with other social networks like Facebook. The system consisted of three major components: data preprocessing, data classification, and data visualization. In the data preprocessing part, Principal Component Analysis (PCA) and Singular-Value Decomposition (SVD) were tested and Backpropagation Neural Networks (BNN) was used to build the model based on the training dataset. The visualization part showed the overall distribution of three different classes of tweets, and also the sentiment against each airline company. A location-based cloud tag was also made in the visualization.

The pro of this paper is they tested two novel dimension deduction method (PCA and SVD) for Backpropagation Neural Networks and achieve reasonable results; Also, they tried to provide both aggregated and fine grain sentiment analysis on airline company which was helpful for end users.

While the con is the system did not contain the data collection part and the test dataset they used is provide, which made the system hard to implement on real business

cases. In addition, the visualization was not automated to maintain for now and manual maintenance is required every time when the data is updated.

There are also a bunch of tools available online which are free or with free trial version for academic or commercial usage. We exclude the paid commercial tools since our proposed architecture will be free to implement as well.

Sentiment viz

Sentiment viz[21] is an online Twitter visualization tool which provides fruitful graphs with different dimension slices. The way it works is like following: Firstly, ask for a keyword that the user is interested in analyzing tweets; Secondly, the system will send a request to Twitter Search API to acquire the most recent tweets based on the keyword; Thirdly, sentiment analysis will be run in the backend, and all the graphs will be updated once the results are decided.

One highlight is the fine-grain analysis the tool can achieve. The sentiment types are based on Russell's model of emotional affect which includes eight sentiments in total like excited, depressed, miserable, etc. For the classification itself, the authors build their own dictionary on top of an extended ANEW dictionary built by researchers McMaster and Ghent Universities [89] and a happiness dictionary built by researchers at the University of Vermont [90] which facilitates their fine grain classification. Also, Sentiment viz provides scatter plot for sentiment overview, topic clusters, tag cloud, timeline, map, and other visualization graphs for the users to dive into.

One concern is although timeline graph is one option, every time a keyword is

[21] https://www.csc2.ncsu.edu/faculty/healey/tweet_viz/tweet_app/

input, only the most recent several tweets will be returned (around 200) which results in a missing part of the historical data. Also, the map graph rarely works, and our guessing is due to the tool is retrieving the location from the user device location which enabled by only 1-2% of the overall Twitter users [85].

Social Searcher

Social Searcher[22] is a tool that helps user to pull the most recent data from several popular social networks including Facebook, Twitter, Instagram, Youtube, Reddit, Flickr, etc. It also requires the user to provide a keyword like Sentiment vis and then pull all the related posts from these different sources. All the post pulled from different sources will be categorized into positive, negative, or neutral category. Apart from the sentiment score for the keyword in terms of different platforms, it also shows the overall volumes and types of posts.

It is a good tool to compare the different sentiment polarity for a specific topic among different social network, especially for marketing people. Considering the eleven sources the tool is pulling from, the response speed is quite fast. Most of the time, the request could be finished within several seconds.

One concern is similar to the one for Sentiment Vis. Social Searcher works in a similar way as Sentiment Vis, and the request only pulls the most recent posts, and the time scope is not controllable by the user. So, there is no way to know how big is this sample among all the available data and how is the current trend compared with two days ago.

[22] https://www.social-searcher.com/social-buzz/

Kintegra Labs

Kintegra Labs[23] is a tool designed for social media marketing and optimization. It has a free trial version which supports sentiment analysis and other functions based on the input keyword by the user. In the dashboard, it will show the overall sentiment distribution on three classes, related keywords, also some other metrics like strength, sentiment ratio, passion, and reach for mainstream social media platforms like YouTube, Twitter, and Reddit. Also, the most relevant post from these sources will be shown in the middle of the dashboard.

The pro of this tool is the visualization of it is clear and straightforward. For example, the way it showcases the post in terms of sources can help people to compare the difference between popular posts from different social media platform.

The con of it is also a lack of historical data. It can pull the most recent data from different sources, but it's hard to track any sentiment trend overtime. Also, there is no explanation about the terms being used on the dashboard like strength, sentiment ratio, and so on, which might cause confusion when people try to understand them.

Talkwalker

Talkwalker[24] is another social media mining tool having free trial for users. It is also a platform that requires a keyword as a start. The metrics on the dashboard includes overall conversation, engagement, sentiment scores, and potential reach. It also supports conversation and sentiment trend visualization over time.

---

[23] https://labs.kintegra.io/dashboard/tools/sentimentanalysis

[24] https://www.talkwalker.com/social-media-analytics-search

One of the highlights of this tool is there are many dimensions users can choose to slice the result. For example, users can filter the result based on media type, location, language, devices, sentiment, and demographic. In addition, it also supports keyword comparison, which facilitates competitor analysis.

One of the cons is the method for keyword searching is unclear since sometimes the two words in the keyword will be separated apart in the search results. Also, there is no description of the metrics which makes it hard to validate the results.

Sentiment Analyzer

Sentiment Analyzer[25] is a free online tool for general purpose sentiment analysis. The interface of it is pretty straight forward: Input several sentences or paragraphs then click on "Analyze Text", then the overall sentiment score will be displayed on the panel located on the right side. The backend classifier is trained based on 8000 writing samples and transcripts of spoken conversations which appear in the American National Corpus (ANC).

The pro of this tool is that it is completely free so that academics or institutions can benefit from it. In addition, the response speed is pretty fast.

There are also some cons of it. Firstly, the result only reflects the sentiment score for the entire provided sentences or paragraphs. Even though it is possible for some sentences to have different meanings, the final score will only show the aggregated score, which means the fine-grain score is not accessible. Also, based on the trait of the training data, it might perform well for similar spoken conversation but not for other ones like news or tweets. Actually, it is pretty hard to build a general-purpose classifier based on one type

[25] https://www.danielsoper.com/sentimentanalysis/default.aspx

of training data.

Senti Strength

Senti Strength[26] is an online tool that estimates the strength of positive and negative sentiment in short texts. It supports different mode of classification type, like binary (positive/negative), trinary (positive/negative/neutral) and single scale (-4 to +4) results. The backend is driven by the Senti Strength lexicon which is a popular tool being used in Twitter sentiment analysis. There are also options for domain selection with six domains and one auto-detect options. Currently, fifteen languages are supported for sentiment analysis.

The highlight of it is it will measure the positive and negative sentiment score parallelly in a sentence level instead of only one aggregated score like most of other sentence-level tools. This is quite useful when a sentence contains more than one sentiment for different aspect like "I like the environment of the restaurant, but the food is bad."

However, most of the research regard SentiStrength as a backend tool or directly use its lexicon since there is no visualization of it available.

SummarizeBot

SummarizeBot[27] is an online text-mining tool to support sentiment analysis functionality. The interface is quite easy to understand. It starts by accepting several sentences or paragraphs as input. After processing, firstly, there will be a document-level sentiment polarity for the entire provided input. Below that section, a sentence level and finer grain sentiment polarity scores will be shown as well. The sentiment orientation is

[26] http://sentistrength.wlv.ac.uk
[27] https://www.summarizebot.com/text_api_demo.html

decided based on the lexicon.

One of the pros of this tool is the easiness for understanding. In the sentence level sentiment visualization, the score of the sentence is shown at the beginning, followed by the sentence with all the sentiment words being highlighted as green for positive or red for negative. This is very easy for users to understand why this sentence is categorized into positive or negative category.

While on concern is the lexicon-based approach is not flexible enough for different domains. In addition, the sentiment will be doubtful when the same word has a different meaning in a different context.

ParallelDots

ParallelDots[28] is also an online sentiment analysis tool with free demo. By inputting your sentences or paragraphs, a sentiment probability score will be displayed for all the three classes: positive, negative, and neutral. The class with the highest probability will be highlighted and enlarged. Currently, 15 languages are supported.

The pro of this tool is the straightforward interface and Multilanguage supports.

One concern is there is only aggregated level sentiment score and the user cannot get access the sentence level score for finer grain analysis. There are also no explanations of how to calculate the probability scores which makes it hard to validate the results.

[28] https://www.paralleldots.com/sentiment-analysis

2.3.2 Gap analysis for sentiment analysis use case studies and online tools

After reviewing several use cases studies and tools, we can see there are still some missing parts from the studies that not being handled very well:

Firstly, there is a lack of data processing mechanism details design about how to apply the proposed models in a daily (or even, hourly) basis for the ones which are expected to drive business products. When applying a proposed model in reality to solve real business problems, there are some other questions to consider in addition to the model design itself. For example, do we want to keep receiving streaming tweets to update our training dataset? How frequently do we want to update our sentiment classification model when some new training data becomes available? When analyzing the tweets, we collected, do we always want to use the newest model, or design a mechanism to use the best model even if it is not the newest one? Do we want to monitor the performance of the different model we built and how? What metrics do we want to show in the dashboard for the stakeholders and why? How frequently do we want to refresh the dashboard? These questions are important to consider when applying any Twitter sentiment analysis pipeline to drive business product, but few of the previous works built a mechanism to take these into consideration.

Secondly, few tools are designed to keep tracing the sentiment trend over time which is important in a real use case. Currently, nearly all the free online tools can only retrieve the most recent 200 tweets after the keyword is provided by the user. Due to the popularity of the keyword, these around 200 tweets could be posted within the past a few hours, or all within the past 30 minutes which means there is no guarantee that a sentiment trend of the keyword would be captured based on the collected 200 tweets. In our opinion, the sentiment trend in Twitter is important for stakeholders to compare the

sentiment orientations for a different time period or find out anomalies from the trend, while this part is still missing from nearly all the free and accessible online tools we reviewed.

2.4 Problem Statement

Based on the gaps mentioned above in terms of the existing Twitter datasets, Twitter sentiment classification models, use cases and tools, we want to define the problem we want to focus on in this study:

For Data Collection: Twitter specific training dataset is the foundation to build a Twitter sentiment analysis model pipeline when using machine learning based approach. While currently, the existing datasets and data collection methods are not sufficient to build the required training datasets for business applications. Firstly, Twitter's policy restricts researchers to republish the collected and labeled Twitter dataset which makes it unrealistic to use the datasets built by others; In addition, even if they are available, using these datasets will suffer from time effect sooner or later; The only way left is to collect your own dataset, while no previous works have used graphic emojis which is easier to be traced and managed compared with string emoticons. To solve these problems, we will propose a data collection method by using Twitter Streaming API which will not be restricted by the redistribution policy, easy to evolve over time, and utilizing graphic emojis as noisy labels instead of string emoticons.

For sentiment classification: Twitter doesn't have any topic limitations, and so as the Twitter datasets that built by noisy labels. While some previous works that using datasets didn't consider any topic limitation feature of Twitter, which leads to a harm in performance since the synonymy and polysemy issues; Although some works using other datasets try to utilize topic models to detect the semantic topics behinds the tweets, there

is no guarantee that their selected topic number is the most proper one or their selected model is the best model. To respond these issues, we will propose a topic-model based hybrid approach design specifically for building a Twitter sentiment analysis pipeline by applying topic modeling as an essential step to uncover the latent topics behind the training datasets. Also, by employing three different topics models with different inputted parameters and topic coherence score, the best topic number and model can be selected in an automated way.

For use case studies and online tools: A group of studies have been done to propose Twitter sentiment analysis models, and a few online tools are built for facilitating sentiment classification tasks. Naturally, it would be helpful if the data processing details about how to apply the proposed model/tool on a daily basis to bring insights to stakeholders in the long term. While currently, the related details for applying the proposed models are missing, and the free online tools have limited ability to trace the historical sentiment trend for the given keyword which decreases the utility of the proposed models/tools. To fill these gaps, we will utilize a use case study to illustrate how to handle the data processing steps to our proposed pipeline when solving a real problem. Steps including data freshness check, model periodically updating, model performance evaluation, front-end dashboard design and refreshing, keyword tracking will be explained in detail.

Chapter 3: Proposed Data Acquisition Method

With the need of proposing a data acquisition method that can solve real business problems or drive product in a daily basis, it is essential to ensure the method has few collecting limitations, has the ability to update with the changing language style on Twitter, and can be built with the least human efforts in terms of labeling and annotating. In the following section, we want to introduce the way we acquire and auto-annotate the training dataset that can meet these needs mentioned above.

3.1 Data Acquisition Method

In our study, Twitter Streaming API[29] is used to acquire the data. Twitter Streaming API would randomly select a subset from the overall tweets generated in nearly real-time. In another perspective, the API could be divided into Standard API and Premium API which would also influence the amount and speed you could acquire the tweets. In our study, we use Premium + Streaming API to acquire data since some of the advanced filter options are only available for Premium API.

A Python script is written to send requests and receive the returned tweets via the Streaming API. A good feature of Twitter Premium API is compared with Standard API, developers could have more flexible options to add filter onto the query to fetch a more useful subset from the random tweets the API selects[30]. In order to build a dataset that could be used for English language applications, the first filter we add is a language filter which only requires English tweets; Also, the existence of retweet could impact the weight of features later, so the second filter we add is only to include the instances that are not

[29] https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data.html
[30] https://developer.twitter.com/en/docs/tweets/rules-and-filtering/overview/premium-operators

retweets; In addition, to make the dataset could be annotated automatically, we apply an emoji filter on our program since emojis could be used as noisy labels. Based on our program, only the tweets that have at least one of the following emojis would be included in the returned tweets.

One of the improvements in our research is that we pick 42 commonly used graphic emojis with positive or negative emotions as our noisy labels, compared with the string emojis that used in previous studies to auto-annotate our training dataset. The detailed graphic emojis used in this study are shown in Table 3. The first reason we choose graphic emojis is they are enjoying their popularity much more than string emojis on Twitter nowadays. Therefore, we expect the tweets returned based on graphic emojis could provide a wider perspective than string emojis. Another benefit by using graphic emoji is they are much more manageable and traceable since there are corresponding Unicode for all graphic emojis so that you can precisely capture any of them by providing a list of Unicode for your chosen emojis. In contract, string emojis could have several different combinations and corner cases that researchers need to deal with carefully. For example, string emoticon with a smiling face could be :), :-), :D, :/D and so on and if a new string emoji is created and used by Twitter users, it is hard for us to be aware of that immediately. If we don't see any people start to use it based on the tweets we have, we won't know that string emoticons is used currently. While graphic emojis are treated as characters these days and there are numbers of websites out there that providing a list of mapping between the graphic emojis and the Unicode. Even if a new graphic emoji is added and available in Twitter or an old one gets removed and no longer available, we can easily check and know it. Overall, we can, and we should utilize graphic emojis as noisy labels to build training dataset in an automated way nowadays.

Table 3 List of Graphic emojis

Unicode	Negative Emoji	Description	Unicode	Positive Emoji	Description
U+1F612		UNAMUSED FACE	U+1F600		GRINNING FACE
U+1F613		FACE WITH COLD SWEAT	U+1F601		GRINNING FACE WITH SMILING EYES
U+1F614		PENSIVE FACE	U+1F603		SMILING FACE WITH OPEN MOUTH
U+1F615		CONFUSED FACE	U+1F604		SMILING FACE WITH OPEN MOUTH AND SMILING EYES
U+1F616		CONFOUNDED FACE	U+1F606		SMILING FACE WITH OPEN MOUTH AND TIGHTLY-CLOSED EYES
U+1F61E		DISAPPOINTED FACE	U+1F609		WINKING FACE
U+1F61F		WORRIED FACE	U+1F60A		SMILING FACE WITH SMILING EYES
U+1F620		ANGRY FACE	U+1F60B		FACE SAVOURING DELICIOUS FOOD
U+1F621		POUTING FACE	U+1F60C		RELIEVED FACE
U+1F622		CRYING FACE	U+1F60D		SMILING FACE WITH HEART-SHAPED EYES
U+1F623		PERSEVERING FACE	U+1F60E		SMILING FACE WITH SUNGLASSES
U+1F624		FACE WITH LOOK OF TRIUMPH	U+1F60F		SMIRKING FACE
U+1F625		DISAPPOINTED BUT RELIEVED FACE	U+1F617		KISSING FACE
U+1F626		FROWNING FACE WITH OPEN MOUTH	U+1F618		FACE THROWING A KISS
U+1F627		ANGUISHED FACE	U+1F619		KISSING FACE WITH SMILING EYES
U+1F628		FEARFUL FACE	U+1F61A		KISSING FACE WITH CLOSED EYES
U+1F629		WEARY FACE	U+1F62C		GRIMACING FACE
U+1F62B		TIRED FACE	U+1F638		GRINNING CAT FACE WITH SMILING EYES
U+1F63E		POUTING CAT FACE	U+1F63A		SMILING CAT FACE WITH OPEN MOUTH
U+1F63F		CRYING CAT FACE	U+1F63B		SMILING CAT FACE WITH HEART-SHAPED EYES
			U+1F63C		CAT FACE WITH WRY SMILE
			U+1F63D		KISSING CAT FACE WITH CLOSED EYES

A list of graphic emojis we use as filter condition is shown in Table 3 with their Unicode, picture, and description. The emojis on the list are chosen since each of them is commonly used in Twitter and conveying clear positive and negative emotions without too much ambiguity. Although the pictures for these emojis could be shown a little different in different devices and systems, the emotions of them are not expected to change significantly. As far as we know, our study is the first one that proposes using graphic emojis instead of string emojis as noisy labels to acquire and label the tweets.

3.2 Data Definition

Twitter API could provide a variety of features in terms of tweet object, user object, entities object, extended entities object, and location object in JSON format. Tweets are the most fundamental ones which are also known as "status updates." The Tweet object has a long list of 'root-level' attributes, including fundamental attributes such as id, created_at, and text and it is also the 'parent' object to several child objects including user, entities, and extended_entities. Tweets that are geo-tagged will have a place child object[31]. Based on the application to be built, people can get nearly find every field they want from the above-mentioned five objects and hundreds of features provided. In our case, with the limited storage space, only some selected fields of all features are collected which are considered sufficient for building a dataset for Twitter sentiment analysis. The features in our dataset are listed in Table 4.

Table 4 Features of Dataset

Features	Type	Definition	Example

[31] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object

tweet_id_str	String	The string representation of the unique identifier for this Tweet.	"tweet_id_str":"105011 8621198921728"
tweet	String	The actual UTF-8 text of the status update.	"tweet": " we will now count all emojis as equal t… https://t.co/MkGjXf9a
tweet_created	String	UTC time when this Tweet was created.	"created_at": "Wed Oct 10 20:19:24 +0000
source	String	Utility used to post the Tweet, as an HTML-formatted string. Tweets from the Twitter website have a source value of web.	"source":"Twitter for iPhone"

hashtags	Array	Represents hashtags which have been parsed out of the Tweet text.
		indices represent an array of integers indicating the offsets within the Tweet text where the hashtag begins and ends.
		text represents the name of the hashtag, minus
			{
			"hashtags": [
			{"indices": [32, 38],
			"text": "nodejs"}
			]
			}

hashtags_num	Int	The number of hashtags in this tweet.	"hashtag_num":1
urls	Array	Represents URLs included in the text of a Tweet. indices represents an array of integers representing offsets within the Tweet text where the URL begins and ends. display_url represents the URL pasted/typed into Tweet. expanded_url represents Expanded version of " display_url".	{ "urls": [{ "indices": [32, 52], "url": "http://t.co/IOwBrTZR ", "display_url": "youtube.com/watch?v
urls_num	Int	The number of urls in this tweet.	"urls_num":1

user_mentions	Array	Represents other Twitter users mentioned in the text of the Tweet.
		id represents the ID of the mentioned user, as an integer.
		id_str represents the ID of the mentioned user, as a string.
		mame represents the display name of the referenced user.
		screen_name represents the screen name of the referenced user.
		indices represents an array of integers representing the offsets within the Tweet text
			{
			"user_mentions": [
			{
			"name": "Twitter API",
			"indices": [4,15],
			"screen_name":
			"twitterapi",
			"id": 6253282,
			"id_str": "6253282"
			}
			]
			}

user_mentions_ num	Int	The number of users mentioned in this tweet.	"user_mentions_num": 1

symbols	Array	Represents symbols, i.e. $cashtags, included in the text of the Tweet.
		indices represent an array of integers indicating the offsets within the Tweet text where the symbol/cashtag begins and ends.
		name of the cashhtag, minus the leading '$'
			{
			"symbols": [
			{
			"indices": [12,17],
			"text": "twtr"
			}

symbols_num	Int	The number of symbols in this tweet.	"symbols_num":1
user_created	String	The UTC datetime that the user account was created on Twitter.	"user_created": "Mon Nov 29 21:18:15
user_description	String	The user-defined UTF-8 string describing their account.	"user_description": "Thinker and seeker."
followers_count	Int	The number of followers this account currently has.	"followers_count": 21

The datatype and definition here are aligned with tweet object[32], user object[33], and entities object[34] which makes it easy to duplicate or re-implement.

3.3 Legal Issues Related to the Dataset

Twitter's data used to be a popular open data source for NLP related tasks. While as the policy of Twitter becomes stricter and stricter, people are not allowed to use and distribute data from Twitter as they used to. There are several conditions for publishing tweets now based on current Twitter's policy.

Generally, data from Twitter could be used for research purpose, but the fields could be shared to the public is quite limited. "If you provide Twitter Content to third parties, including downloadable datasets of Twitter Content or an API that returns Twitter Content, you will only distribute or allow download of Tweet IDs, Direct Message IDs, and/or User IDs.[35]" This is also the way how organizers of SemEval release their datasets for researchers. Since the fields that could be shared are very limited, it won't be worthwhile for us to distribute our dataset; Also, it might be a better option for people to always acquire the most recent data from Twitter for their own research so that the data freshness won't be an issue; Furthermore, the usage of noisy labels and filters we introduced in our study can be duplicated and reimplemented easily so that people can always refer to our list of emojis to get their own data.

3.4 Discussion

In this section, we want to discuss some considerations when building our

---

[32] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object
[33] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/user-object
[34] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/entities-object
[35] https://developer.twitter.com/en/developer-terms/agreement-and-policy.html

proposed data collection method in our pipeline.

First of all, we are motivated to build and use this method to always collect most updated data which is due to the changing language style among tweets. In social media like Twitter, one of the most different features compared with other platforms like e-commerce or news websites it the changing language style over time. For the datasets about customer review or news, the language style is more formal and more static over time than the style in Twitter so that the time effect is not as concerned as it is in Twitter sentiment analysis. While in Twitter, new buzz words, hashtags, slangs are being created every day and evolve rapidly over time, which means the Twitter specific dataset that was valid when being used to build a model 5 years ago could no longer be valid today, since the training dataset doesn't contain the most recent tweets with the most updated language styles in the test dataset and leads to a poor performance. Therefore, we propose that in a real use case, the pipeline will be utilized to analyze the most updated tweets, so that the training dataset needs to be updated periodically as well. That is the reason why we always need more updated training dataset over time.

In addition, the usage of Twitter Streaming API and emojis are essential in our method design to make the pipeline as automated as possible. When we are talking about automated sentiment analysis pipeline, we refer to the pipeline needs to have the ability to take in the newest data when it is available, utilize the data to re-build the classification model, and evaluate and apply the best model on the test dataset. Therefore, a data collection method which can keep receiving and automatically labeling the acquired data as positive or negative class is the key to drive the whole pipeline automated. So far, it is clear that why we choose to utilize Twitter Streaming API, and why we build our own graphic emoji list. The former one is to meet the need of collecting most updated tweets;

The latter one is used to auto-categorize the collected tweets into positive or negative class. By doing this, we want to minimize the human effort in this process and ensure our data collection is run in an automated way.

Last but not least, we propose that building our own graphic emoji list is better than using string emoticons which is proposed in previous works. Our first consideration is that graphic emojis are much easier to be traced and searched, which makes it a better candidate for noisy labels. Because of the popularity of emojis, nowadays each of them has its own Unicode across different platform, which means you can search for one just like searching for any character. A variety of online sources can be found about the categorization of emojis[36] [37], and some webpages are also built to monitor the live popularity for all the emojis[38]. Therefore, using graphic emojis means that you can have a complete scope about what are all the emojis that are available on Twitter (also for any other social platform), and what is the overall volume, increasing speed, or popularity of a specific emojis. By knowing these, you can decide the collection of emojis for your use case or estimate whether the volume of selected emojis is big enough for building your dataset. In contract, using string emoticons has none of the above-mentioned benefits. Neither can you know how many string emoticons are being used by people at this moment (you can make estimations, but it is hard to exhaust all the probabilities) because there is no standard Unicode for them, nor can you estimate the volume and increasing speed of these emojis. Due to all the benefits of using graphic emojis rather than string emoticons, and there is no previous work that explicitly used graphic emojis as noisy

---

[36] https://emojipedia.org/unicode-10.0/
[37] https://unicode.org/emoji/charts/full-emoji-list.html
[38] http://emojitracker.com/

labels, we decided to make our own graphic emoji list and apply it as a filter when collecting our training data.

Chapter 4: Proposed Solution for Sentiment Classification

In the previous chapter, we introduce the way we collect data for our proposed system. In this chapter, we will propose a topic-model based hybrid sentiment classification model for Twitter, based on the data we collected. By proposing this model, our goal is to capture the latent topic behind the tweets and improve the classification performance.

4.1 Overall Architectural Design

In this section, we are going to give an overview of the high-level architecture of our proposed topic-model based hybrid sentiment classification approach.

As shown in Illustration 2, our system consists of three major sections: Data Preparation, Classifier Building, and Evaluation. We mention Data Acquisition here for consistency purpose includes all the steps needed to be done to acquire the data from Twitter which is introduced in chapter 3. Based on the data we get in the previous section, Data Preparation is the process to run a data preprocessing, data resampling and feature selection to build a clean dataset that could be fit into the model later. The third process is Classifier Building which is the core part of the system that firstly tries to utilize the topic model to get the underlying topics behind the tweets, and then clustering based on the topics and building ensemble classifiers in terms of different clusters. Last but not least, the Evaluation will measure the performance of the classifiers and compare our proposed solution with other works.

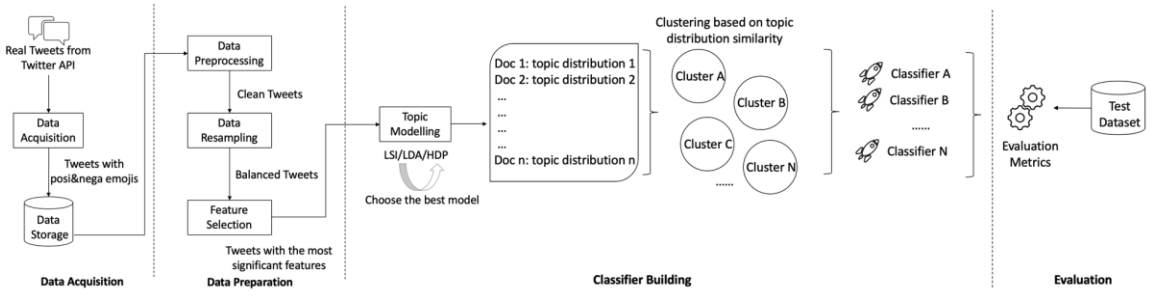


Illustration 2 Architecture of the sentiment classification system

About the overall architecture, we want to clarify why topic modeling is necessary and why our approach is named a hybrid approach. The reason why we want to integrate topic modeling into our pipeline is determined by the way we collect and auto-label our training data based on emojis which is introduced in Chapter 2. Because we are acquiring the training data based on graphic emojis, there will be no domain limitation on the collected tweets. If we use these tweets to build a sentiment classification model, it may suffer from poor performance due to the existence of synonymy and polysemy in the training corpus. In order to minimize the error in classification caused by a lack of domain limitation, we apply topic modeling as the first step after data preparation to capture the latent semantic topics behind the training dataset. For the reason that we call our proposed method a hybrid approach, it is because we combine unsupervised learning approaches (topic modeling, K-means clustering), supervised learning approaches (Random Forest and Logistic Regression), and lexicon knowledge (the usage of Lexicon features) together in our method. As a convention from the previous works, and as it is defined in Chapter 1.2, we name our proposed method a hybrid sentiment analysis approach. More details about each process will be discussed in the following chapters.

4.2 Data Preparation

The data preparation aims at converting and transforming the dataset we get to a cleaner dataset that can fit into the following model. The process is essential because the quality of the built model and analysis are all based on this step. In this part, we aim at dealing with different problems when cleaning the data, including the preprocessing, resampling, and feature selection.

4.2.1 Data preprocessing

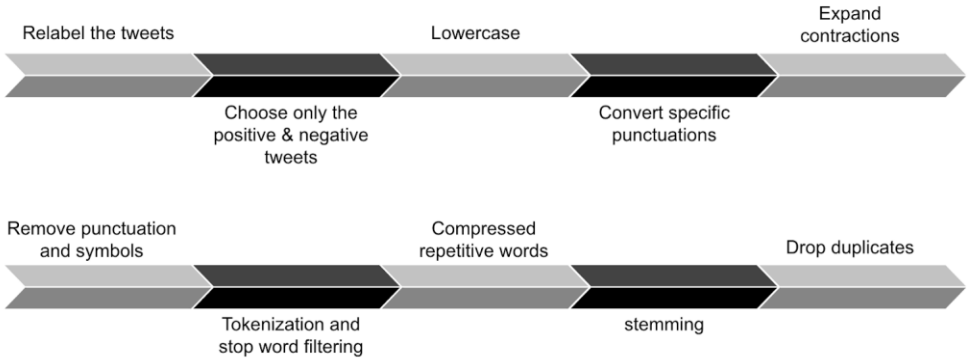


Illustration 3 Data Preprocessing Steps

Text data are known that more likely to contain noises compared with numeric data since there is no strict standard for people when they are expressing their feeling on social media. This problem specifically needs to be considered when we need to deal with text data from Twitter. To get clean data for further analysis, a rule-based data preprocessing is being used to run fundamental cleanup to get rid of obvious noises from the data which is illustrated in Illustration 3. The preprocessing steps included in this study are:

1. **Relabel the tweets**

Although we set the rule to draw tweets only contains positive or negative emoji based on the list we provided, there still some tweets which could consist of both positive and negative emojis inside them. In that case, those tweets won't be regarded as valid tweets for our training dataset. To decrease the noise for further analysis, it is essential to relabel the tweets to distinguish the ones with only positive emojis, only negative emojis, and the ones with both sentiment emojis.

2. **Take only the positive and negative tweets and labels**

Based on what we did in the previous step, only the ones with clear sentiment orientation, which means either positive emoji or negative emoji could be found in the

tweet would be saved. The ones that have two conflict emojis would be filtered out.

3. Lowercase all the tweets

This step aims at making all the letters appeared in the tweets lowercase so that the same words with either uppercase or lowercase would be considered the same word which could help to decrease the number of features and also reduce the dimension when building the model later.

4. Convert specific punctuations

When getting tweets from Twitter's API, there was some encoding/decoding issue between Unicode and the code you might want to use like 'utf-8' or 'ASCII'. We can get a variety of commonly used punctuations like '"','-', or ''' especially ''' which might be a part of contractions but written in Unicode. Therefore, we want to deal with these Unicode symbols carefully to not lose any information from them. We run a special step to convert these specific punctuations into 'utf-8' for further parsing.

5. Expand contractions

We collect a list of commonly used 118 contractions in real life as a reference list (E.g. "ain't" is mapping to "am not"). For each tweet, if any of the contraction appears in the tweets, the contraction will be expanded based on the reference list. The step is taken before we get rid of the punctuations and symbols since we want to keep as much information as possible. Also, it's not rare that we put negations within contractions. Therefore, it's essential to extract them out from the contractions so that we won't lose any important features.

6. Remove punctuations and symbols

For each tweet, all the URLs, html tags, numbers, punctuations, the @username, and #hashtag are dropped from the tweets to get a clean list of texts.

7. Tokenization and stop word filtering

Tokenization refers to the process which breaks down a document or a sentence to pieces of individual words. Based on the bag-of-word assumption, tokenization is an initial step to turn the word into a vector space model for modeling.

8. Compress Repetitive Words

Since the language style in Twitter is quite casual so that sometimes words like 'goooood' which people made up to show the intensity of their feelings would appear frequently. This step aims to convert any word that contains three or more than three same letters in a row within the word to two same letters. For example, 'goooooood' will be converted to 'good'.

9. Lemmatizing or Stemming

Lemmatizing or Stemming are commonly used ways to group words with the same root together. For example, 'good' is the lemma of 'better' and 'best'. Since good, better, and best all have positive meanings with different degree, they could work similarly in a sentiment analysis task, but now they are three different features. By lemmatizing, we can group these kind variants with the same lemma to the same way which can help us significantly reduce the amount of the features.

10. Drop duplicates

It is possible to have duplicated tweets after the above process. Therefore, in the very end, we want to drop all the duplicates so that no tweet would be weighed more than it is supposed to be.

4.2.2 Data resampling

From our observation of the data we acquire via Twitter API, the number of tweets with positive emojis is always greater than the amount with negative Emojis (from the raw

data we draw, the size of positive ones could double the negative ones). From paper [49], [91], imbalanced data could introduce some potential problem on the classification result. Therefore, data resampling is an essential part of data preparation to get a more balanced dataset that we can use for classification later.

In our study, we apply the Random under sampling method since its simplicity and has been proved to achieve good performance in [49].

4.2.3 Feature representation

In this section, we will describe the features we use to build the model including N-gram features, lexicon features, negation features, and POS features.

N-gram features

Tweets could be regarded as any document when applying text mining technique on it while one thing that cannot be ignored is that the noisy and short nature of tweets. We try to deal with the noisy part in the previous section while the short trait is something we need to handle too. In this study, N-gram model is being used for feature presentation. To deal with the short nature of tweets, the performance of unigram, unigram, and bigram feature will all be tested due to the inclusion of trigram will significantly increase the dimension space while only using unigram might not capture all the valuable features since some of the tweets tend to be very short. Another reason to use N-gram features is it will dynamically evolve over time as we consistently get new training data into the model compared with using only lexicon words as features.

After applying N-gram model on data, word count and tf-idf representation approaches will be applied to add the weight to the vector space. In previous works, tf-idf usually outperform word count so that we choose tf-idf as our feature representation method.

Lexicon features

In some previous study, lexicon features are proved clearly increase the accuracy and performance when applying machine learning model. Although lexicon knowledge might not be flexible enough in some scenarios, they still could be regarded as part of the features to help to make a better classification.

The lexicon we use in this study is SentiWordNet 3.0 [37] which organizes more than 100k words and sentiment scores. For each word in the lexicon, it will be assigned scores in terms of the extent of positive, negative and subjective which would add up to 3. For example, the word "able" has a positive score of 0.125, a negative score as 0, and a subjective score as 1 - 0.125 = 0.875. In SentiWordNet 3.0 [37], the same word could have different meanings in different contexts (which is explained in the Gloss column). While for simplicity, we take the average of the positive and negative sentiment scores for the same word and take them as the final scores for the word. Then for each tweet, we will build the following lexical features:

- sum_senti_score: The summary of the sentiment score for a tweet;
- sum_senti_pos_score: The summary of the positive sentiment score for a tweet;
- sum_senti_neg_score: The summary of the negative sentiment score for a tweet;
- pos_neg_score_ratio: If negative sentiment score is not zero, it equals to sum of the positive sentiment score / sum of the negative sentiment score. Otherwise, set it to zero. The metric would always between 0 and infinite which could indicate the extent that positive sentiment bigger than the negative sentiment.
- pos_word_ratio: If tweet length (the number of tokens) is not 0, it equals to the amount of positive words in the sentence with a score that higher than 0.3 divided

by the tweet length. Otherwise, set it to 0. This metric could help to indicate the ratio of positive words account for the overall tweet.

- neg_word_ratio: If tweet length (the number of tokens) is not 0, it equals to the amount of negative words in the sentence with a score that higher than 0.3 divided by the tweet length. Otherwise, set it to 0. This metric could help to indicate the ratio of negative words account for the overall tweet.

Negation features

Handle negation appears in the tweets is another consideration of us when building a robust model since the use of negation will potentially reverse the sentiment polarity of the tweet. To consider the impact of negation, we make a list of negation words, e.g., “no”, “not”, “never”, “rarely”, etc. For each tweet, we will count the number of negation words that appear in the tweet, divide by 2, and then take the remainder of it. In this way, the value of this column will always be 1 or 0, and the idea behind this is if the number of negation words is odd then the sentiment polarity will be reversed while when the number is even then the sentiment orientation could stay the same way.

POS features

POS (part-of-speech) is also included as part of our features. We mainly consider the ratio of words with the following 5 POS tags: Adjective, Adverb, Noun, Pronoun, Verb. For words with POS tags in 'NN', 'NNS', 'NNP', 'NNPS' would be regarded as Noun; For words with POS tags in 'VB', 'VBD', 'VBG', 'VBN', 'VBP', 'VBZ' would be regarded as Verb; For words with POS tags in 'JJ', 'JJR', 'JJS' would be regarded as adjective; For words with POS tags in 'RB', 'RBR', 'RBS' would be regarded as adverb; For words with POS tags in 'PRP', 'PRP$', 'WP', 'WP$' would be regarded as Pronoun; for words that not in any of the above mentioned groups would be regarded as others.

For each tweet, we calculate the following ratio:

- adj_ratio: If overall tweet length is not 0, it equals to the number of adjective appears in the tweet / overall tweet length. Otherwise, set it to 0;
- adv_ratio: If overall tweet length is not 0, it equals to the number of adverb appears in the tweet / overall tweet length. Otherwise, set it to 0;
- noun_ratio: If overall tweet length is not 0, it equals to the number of noun appears in the tweet / overall tweet length. Otherwise, set it to 0;
- pronoun_ratio: If overall tweet length is not 0, it equals to the number of pronoun appears in the tweet / overall tweet length. Otherwise, set it to 0;
- verb_ratio: If overall tweet length is not 0, it equals to the number of verb appears in the tweet / overall tweet length. Otherwise, set it to 0;
- other_ratio: If overall tweet length is not 0, it equals to the number of part-of-speech that other than the above-mentioned ones that appear in the tweet / overall tweet length. Otherwise, set it to 0;

4.2.4 Feature selection

We want to run a feature selection for the N-gram features specifically since the vector space is pretty big for N-gram features and any only part of them are helpful for the model building. Based on our data exploration, we could see that a variety of infrequently used words could be seen in the dataset (88.3% of the words are being used less than 10 times while only 1.87% of the words is used more than 100 times across all the tweets) which might introduce some noise into the analysis. In order to reduce the impact of some rarely used words, we want to run a feature selection step to filter out the top n most representative ones to build our model later. Chi-square [92] method is selected for our feature selection. The way to calculate chi square is:

$$X^2 = \sum \frac{(O-E)^2}{E}$$

where O refers to the observed frequency, while E refers to the expected frequency if no relationship existed between the variables. After applying Chi-square test, only the top n (a parameter that provided by user) most statistically significant features will be kept in the matrix, which can help to reduce the running time of the later topic modeling, and also drop the commonly used words in both positive and negative classes, which can bring some noise later.

4.3 Topic Modeling Prerequisites

In this section, we will discuss some topic modeling prerequisites for the later model building. One of the limitations of using lexicon-based methods or traditional machine learning methods for sentiment analysis is they cannot capture the semantic meaning behind the text. When they are modeling the text, what they can utilize are the words themselves and maybe documents. While when it comes to deal with polysemy and synonymy, none of them have satisfying performance. Polysemy is to describe the cases that the same word has several different meanings. For example, “man” can be used as “man” as the opposite to “woman” or imply human being when compared with “animals” or means buddy in oral English. In this case, “man” is always used as a noun even it could have different meanings in different scenarios. Actually, some words could have different meanings and even different part-of-speech which may make the situation more complicated. Another concern, synonymy, indicates a group of words is actually implying the same meaning. An example of this could be the words “buy” and “purchase” can indicate a very similar meaning and be exchangeable in most scenarios. The existence of synonymy also brings some challenges in sentiment classification tasks. Lexicons could be

good resources and knowledge base, but themselves don't really help to solve the polysemy and synonymy problems. Traditional machine learning approaches are usually using bag-of-word model which means they can get acceptable performance for the words they have seen but not for the words they have never seen before but having the same meaning with the previous words since it is still based on words but not semantic. To solve all these issues, we want to integrate topic modeling in our sentiment classification pipeline to capture the hidden semantic meaning behind the text and boost the performance of our classifier later.

Topic models are used to identify the hidden semantic topics from a collection of text and find short descriptions of the members which can enable efficient processing of large collections while preserving the essential statistical relationships that are useful for basic tasks such as classification, novelty detection, summarization, and similarity and relevance judgments [93]. The fundamental assumption behind all the topic models is each document consists of several different topics, and each topic consist of a group of different words. When given a collection of documents, the observed parts are the documents and words, while it is believed the latent topics, which carry the semantic meanings, governed the process of generating these documents. And the goal of topic modeling is to uncover these latent topics and help people to understand better about the relationship between topics, documents, and words. In the following sections, we will introduce some popular topic models, and some of them will be used in our architecture later.

4.3.1 Latent Semantic Indexing (LSI)

LSI [94] is one of the earliest topic models that was proposed to uncover the hidden semantic topics behind words. When given a large collection of N documents, they would be represented in vector space in terms of word count of tf-idf (term frequency - inverse

document frequency) for further calculation. After building the vector space matrix which has N rows and M words (N is the amount of records and M is the unique word list from the corpus or from a dictionary), a mathematical approach called singular value decomposition (SVD) will be used to decompose the matrix into three pieces: document-topic matrix U_t, topic strength matrix S_t, and topic-word matrix ${V_t}^T$ which is shown in Illustration 4.

$$A \approx U_t S_t {V_t}^T$$

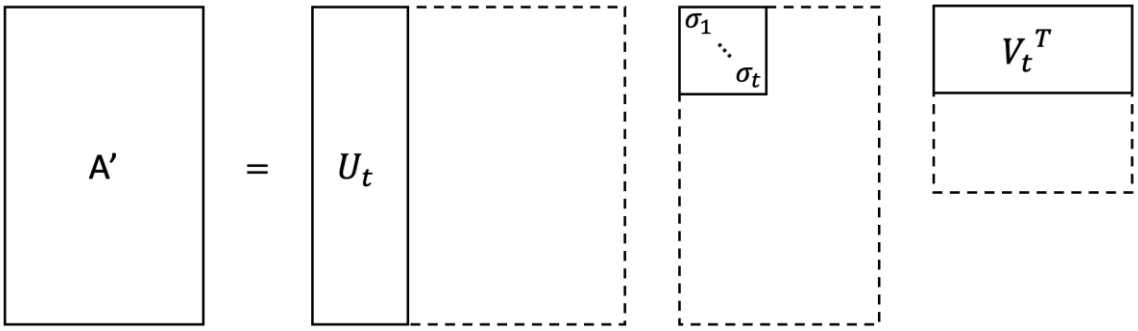


Illustration 4 Singular Value Decomposition Explanation

In this way, the latent structure of the documents would be uncovered. For example, we can zoom in to check which documents belongs to which topic from the document-topic matrix and also, we can measure the similarity between two or more documents by their topic labels. Moreover, it is clear that what are the most commonly used words under a specific topic by examining the topic words matrix. One more benefit from this is the dimensions of the text data are significantly reduced.

4.3.2 Probabilistic Latent Semantic Indexing (pLSI)

pLSI [95] is a topic model that use statistical method to find out the hidden topics behind the collections. As an updated version of LSI, pLSI utilize a probabilistic method other than SVD to solve the problem and also make the model more generative. With the same assumption of any other topic models, pLSI treat the probability of each co-

occurrence as a mixture of conditionally independent multinomial distributions.

The process of generating the documents are:

1. For the document, choose the topic for the document based on P(z|d);
2. For the topic, choose the word from the topic based on P(w|z);

In this way, the likelihood of observing these topics and words are modeled. A pLSI generating process is shown in Illustration 5.

$$P(w,z) = \sum_{d} P(d)P(z|d)P(w|d) = P(z)\sum_{d} P(d|z)P(w|d)$$

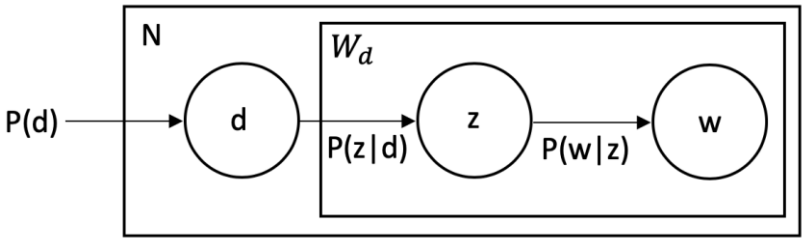


Illustration 5 pLSI Process

In which w refers to a word, d refers to a document, and z refers to a topic. P(d) is a value that could be observed from the corpus directly. While P(z|d) and P(w|z) are calculated based on multinomial distributions which could be modeled by using expectation-maximization algorithm (EM). Actually, the three components P(d), P(z|d) and P(w|z) can be regarded as the corresponding parts as the topic matrix, document-topic matrix, and topic-word matrix but generated based on a probability model. pLSI is a more flexible model compared with LSI, but it also has some issues. One of them is there is no generative probabilistic model for the documents so that it might be problematic to assign a probability to new documents that are not shown in the training set.

4.3.3 Latent Dirichlet Allocation (LDA)

LDA [96] is one of the most popular topic models which could be regarded as an

improved version of pLSA that utilizes Dirichlet distribution to help to generate the document-topic and topic-word matrices. The usage of two Dirichlet distributions to infer the document-topic distribution and topic-word distribution makes it a more generic model than pLSI and could be applied on new documents easily.

The process of generating the documents is shown in Illustration 6.

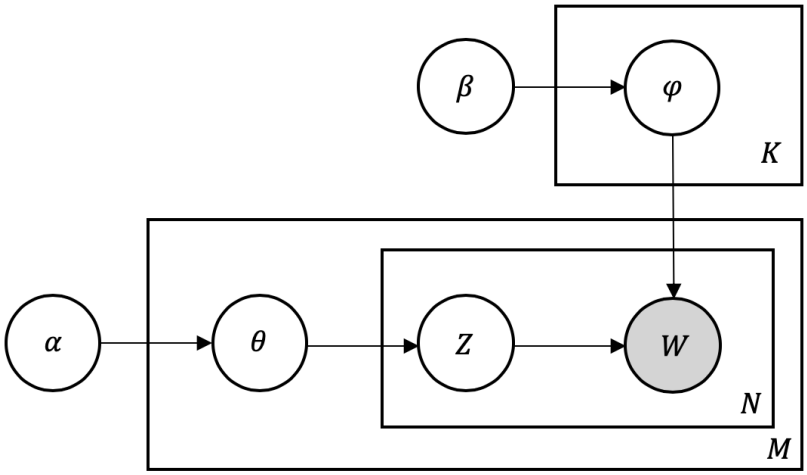


Illustration 6 LDA Process

1. From a Dirichlet distribution Dir(α), a topic distribution θ would be drawn for a particular document n;
2. From topic distribution θ, a particular topic Z is selected for document n;
3. From a Dirichlet distribution Dir(β), a topic-word distribution φ would be drawn for topic Z;
4. From topic-word distribution φ, the final word w would be selected;

Given the parameters α and β, the joint distribution of a topic mixture θ, a set of N topics z, and a set of N words w is given by [96]:

$$p(\theta, z, w|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N} p(z_n|\theta)p(w_n|z_n, \beta)$$

LDA is the most popular topic models which can lead to interpretative results. From

the output, a document-topic distribution could be expected which shows the probability that how likely the document is related to the topics; Also, a topic-word distribution could be expected too which illustrates how likely the words associated to each topic.

4.3.4 Hierarchical Dirichlet Process (HDP)

We can see how LSI, pLSI, and LDA evolves to overcome the problem of the previous model. However, the above three models all require the number of topics as one of the inputs. This is not an issue when people already have a deep understanding of the corpus while it would be much more challenging when that is not the case. Therefore, Hierarchical Dirichlet Process [97] (HDP) model is proposed to address a similar problem but when the number of topics is unsure. As the name infers, the number of topics is generated by a Hierarchical Dirichlet Process which makes it a variable of the model rather than a constant. The advantage of using HDP is there is no limitation of topic number which could even go to infinite that make it a more flexible model; While the downside could be sometime the results are not really understandable especially when the topic number is high, and the data is messy.

4.4 Topic Model Based Classifier Building

In this section, the process of building a sentiment classifier based on topic modeling will be described in detail.

4.4.1 Topic modeling pipeline

So far, we have introduced the most commonly used topic models. Ideally, the most recently developed models tend to achieve better results than the older ones. While since tweets are special kind of documents which tend to be short, informal, and messy, so we cannot guarantee that one particular model would outperform other ones in all scenarios. Therefore, our approach is to build a number of topic models based on LSI, LDA and HDP

models with different topic numbers (except for HDP model), evaluate the performance of all the models and choose the model that outperform any other models.

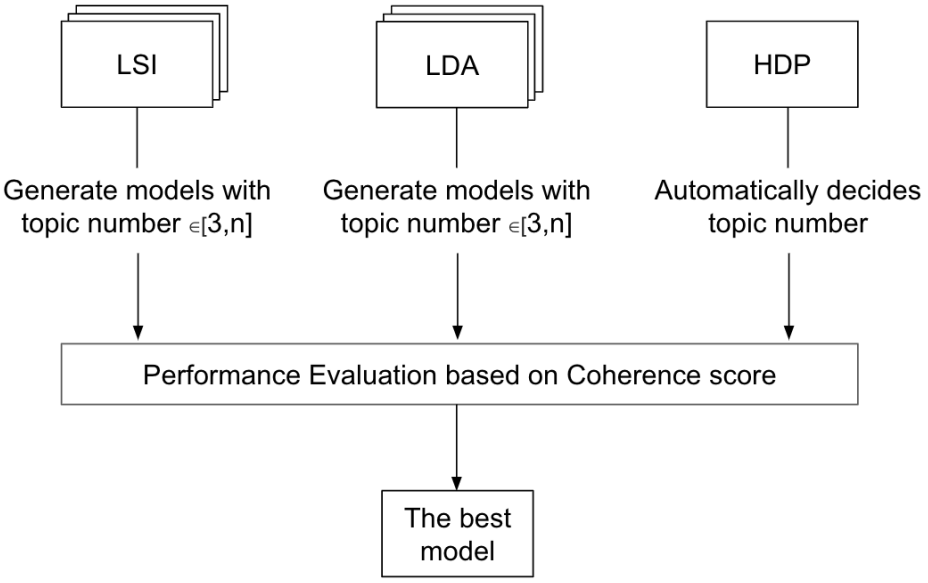


Illustration 7 Topic Modeling Pipeline

The topic modeling pipeline is shown in Illustration 7. For the training tweets, we build LSI and LDA model in terms of different topic numbers ∈ [3, n] (n=30 in this study) which we generate 28 models for LSI and LDA models. We don't need to iterate the topic numbers for HDP models since the topic number is auto decided. After we have all the topic models on hand, we will fit them all into a performance evaluation step to choose the best topic model among all the models. This is also the model that will be used in further calculation.

There is not a universally good metric to evaluate the performance of topic models since even for humans, the comprehension for which is good, and which is bad could be different and subjective. Generally, some of the existing methods that are being used include:

Eye Balling Models

Eye balling models are used in [98], [99], [100]. These methods which provide some convenient ways for visualizing the document-topics and topic-words matrices to help people make a better and faster judgment.

Perplexity

Perplexity [101] is a metric usually being used to evaluate how good the prediction is made by a probability model for a specific sample. Formally, the equation is:

$$2^{H(p)} = 2^{-\sum_x p(x) log_2 p(x)}$$

where $H(p)$ is the entropy of the distribution and x refers to the scope of events.

While as a measure for topic models, perplexity is not a good one since it can only compare the model and new test data in a word-to-word level but not a semantic level. In other words, topic models aim at capturing the latent topic and semantic meaning behind the corpus while perplexity is not really helpful for semantic level evaluation.

Topic coherence

Topic coherence is a group of metrics discussed in [102], [103], [104], [105], which aims at measuring how semantically coherent the model is. Unlike perplexity, it doesn't only compare the word from model and test data but considers some context from the training corpus as well which make it a better measure of semantic coherence. Based on [105], some commonly used metrics for topic coherences are C_{UCI}(based on pointwise mutual information or PMI), C_{UMass}(an asymmetrical confirmation measure between top word pairs or smoothed conditional probability), and C_v (based on context vectors for every topic's top words). We choose using C_v here since it would always drop within [-1, 1] when metric like C_{UMass} tend to fluctuate among [-∞, +∞] which is not so easy for comparison.

The equation for C_v is:

$$v_{ij} = NPMI(w_i, w_j)^{\gamma} = \left(\frac{log\,\frac{P(w_i, w_j) + \epsilon}{P(w_i)P(w_j)}}{-\log\left(P(w_i, w_j) + \epsilon\right)}\right)^{\gamma}$$

In which

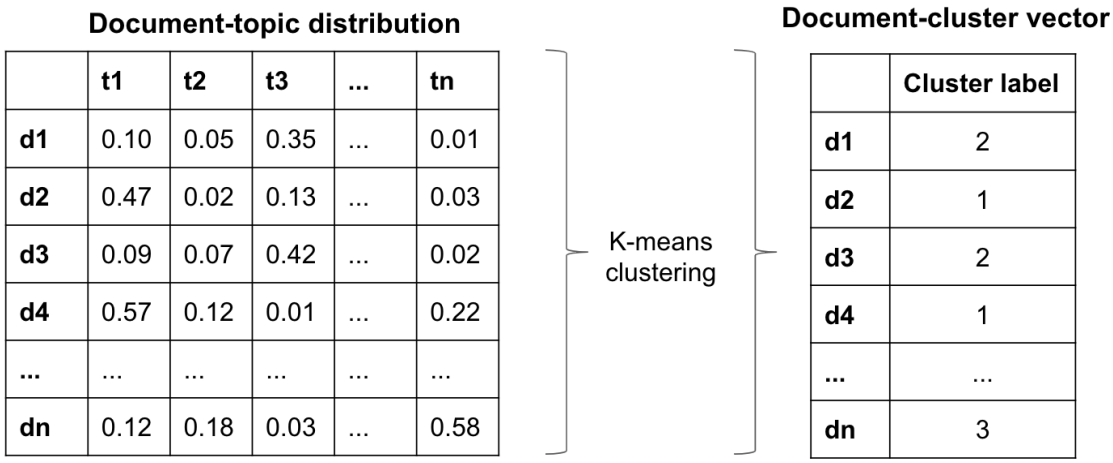
$$\mathrm{P}(w_i, w_j) = \frac{Number\ of\ documents\ that\ contain\ words\ w_i\ and\ w_j}{total\ number\ of\ documents}$$

$$\mathrm{P}(w_i) = \frac{Number\ of\ documents\ that\ contain\ words\ w_i}{total\ number\ of\ documents}$$

$$\mathrm{P}(w_j) = \frac{Number\ of\ documents\ that\ contain\ words\ w_j}{total\ number\ of\ documents}$$

Among these three methods, the ones require human judgments in the middle is not a good option for our system since as long as there is update on training corpus the human judgments would be required again for the new model. In addition, perplexity is not a good measure for topic models since it cannot really capture the semantic level information which is the goal why we want to build topic models. Therefore, the final performance metric we choose is C_v which is one of the topic coherences models that aims at measuring the semantic coherence, and easy to compare with other models since its value would drop between -1 and 1.

4.4.2 Clustering based on topic distribution



	t1	t2	t3	...	tn
d1	0.10	0.05	0.35	...	0.01
d2	0.47	0.02	0.13	...	0.03
d3	0.09	0.07	0.42	...	0.02
d4	0.57	0.12	0.01	...	0.22
...	...	...	...	...	...
dn	0.12	0.18	0.03	...	0.58

	Cluster label
d1	2
d2	1
d3	2
d4	1
...	...
dn	3

Illustration 8 Clustering based on Topic Distribution

After applying topic modeling, one of the outputs from the model is the document-topic distribution matrix which describes the probabilities about how each document belongs to each topic. From this document-topic distribution matrix, we can see how different tweets drop into a different semantic category. For example, assuming there are four topics in total, then a tweet with *p(t) = array([0.10, 0.10, 0.10, 0.70])* will obviously more incline to the fourth topic since the probability is 70%; In contract, tweet with p(t) = array([0.80, 0.10, 0.05, 0.05]) will more likely talking about the first topic.

While those are all the good cases since sometimes one tweet could have high probabilities on more than one topic, or the probabilities are evenly distributed which makes it hard to determine which is the most dominating topic. Due to all these scenarios could happen on our data, instead of choosing the topic with the highest probability as the topic for the tweet, we choose to use a clustering technique to categorize the tweets based on the document-topic distribution to consider all of these cases. In this way, tweets with similar distributions would be assigned into one category even with some noises. By doing

this, we can make the system more resilient and also consider all the information from the topic modeling step.

The output of this part will be a document-clustering labels vector which contains the clustering label for each document. Later on, we will use these clustering labels to construct separate classifiers. The whole process is illustrated in Illustration 8.

4.4.3 Classifier building based on clustering groups

After the clustering in the previous step, we now have the cluster labels – tweets mapping for every tweet in training dataset. In this section, we will build separate classifiers based on two machine learning methods as a final step for model training. This step is shown in Illustration 9.

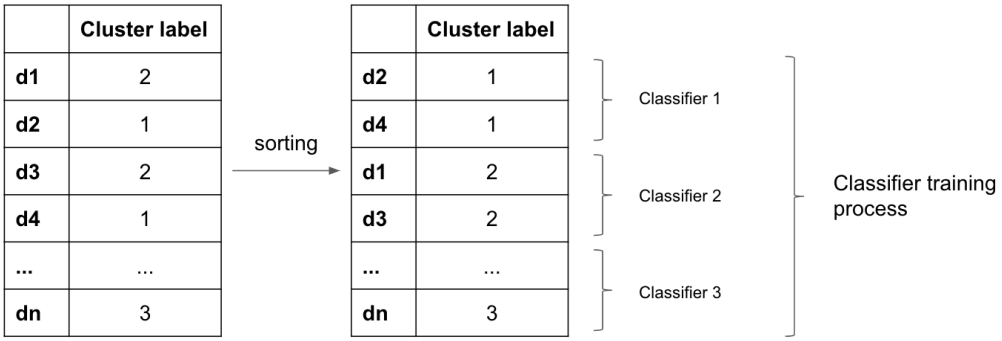


Illustration 9 Classifier Building based on Clustering Groups

4.4.3.1 Random Forest classifier

Random Forest [106] is an ensemble learning algorithm based on decision tree. It is designed to solve the problem that the classifier tends to overfit when growing quite deep which will lead to high variance. By taking the average of several decisions that are growing deep separately, the variance could be reduced. The ensemble approach drives a Random Forest algorithm is bagging. Each time, a subset of the training data would be

selected and used to train the base tree learners and this process would be repeated several times. After several tree learners are built and several prediction or classification is made, an average or majority voting could be applied to take the final result. In this study, we use the average of the probability from each tree instead of the vote from each individual classifier as the final result.

4.4.3.2 Logistic Regression classifier

Logistic Regression is also a popular model that is used widely in text classification. It is firstly proposed in [107] and then start to become a general statistic model on different domains. It is called Logistic Regression since the probabilities of the outcomes are modeled by a logistic function.

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

Despite the fact that it is called "regression", it is a classification algorithm that could be used for binary classification or extended to multi-classification problems.

4.5 System Structure Code

Overall, the pseudo-code of building the sentiment classification model introduced in this chapter (after data preprocessing) is shown in Table 5.

Table 5 Pseudo Code of Classifier Building

Algorithm: Sentiment Classification Model Building
Input: training_dataset *Train* training_dataset_labels *Y* test_dataset *Test* test_dataset_labels *Y_test* lda_min_topic_number *LDA_min* lda_max_topic_number *LDA_max*

lsi_min_topic_number *LSI_min*

lsi_max_topic_number *LSI_max*

min_cluster_number *Kmeans_min*

max_cluster_number *Kmeans_max*

sampling_method *Sampling_mode*

feature_extraction_mode *FE_mode*

feature_represent_mode *FR_mode*

feature_selection_mode *FS_mode*

top_feature_numbers *top_n*

classifier *clf*

feature_mode *Feature_mode*

Output:

classification report (recall, precision, accuracy, F-measure)

Procedure:

train topic models

LDA_model_list = train_model('LDA', *LDA_min, LDA_max*)

LSI_model_list = train_model('LSI, *LSI_min, LSI_max*)

HDP_model = train_model('HDP')

best_LSI_model = choose_best(LSI_model_list)

best_LDA_model = choose_best(LDA_model_list)

best_topic_model = topic_coherence_evaluation(best_lsi_model, best_lda_model, hdp_model)

add clustering information to training matrix

resampled_Train = data_resampling(*Train*)

doc_topic_distribution = best_topic_model.apply(resampled_Train)

doc_topic_distribution['cluster'], best_cluster_num =

Kmeans_choose_best(document_topic_distribution, *Kmeans_min, Kmeans_max*)

feature selection

selected_Train = feature_resampling(*Sampling_mode*)

selected_Train = feature_extactor(doc_topic_distribution).apply(*Feature_mode*)

selected_Train = feature_representation (*FR_mode*)

```
selected_Train = feature_selection(FS_mode, top_n)

# classifier building
for i in set(doc_topic_distribution['cluster']):
    clf = clf.fit(selected_Train.where(cluster label == i), Y.where(cluster label == i))
    clf_list.append({i: clf})

# evaluation based on test data
test_ doc_topic_distribution = best_topic_model.apply(Test)
test_ doc_topic_distribution['cluster'], best_cluster_num =
Kmeans(test_document_topic_distribution, best_cluster_num)
selected_Test = feature_extactor(test_doc_topic_distribution).apply(Feature_mode)
selected_Test = feature_representation(FR_mode)
for i in set(test_ doc_topic_distribution ['cluster']==i):
    Y_pred_piece = clf[i].predict(selected_Test.where(cluster label == i), Y_test.where(cluster
label == i))
    Y_pred.append(Y_pred_piece)
return performance_evaluation(Y_pred)
```

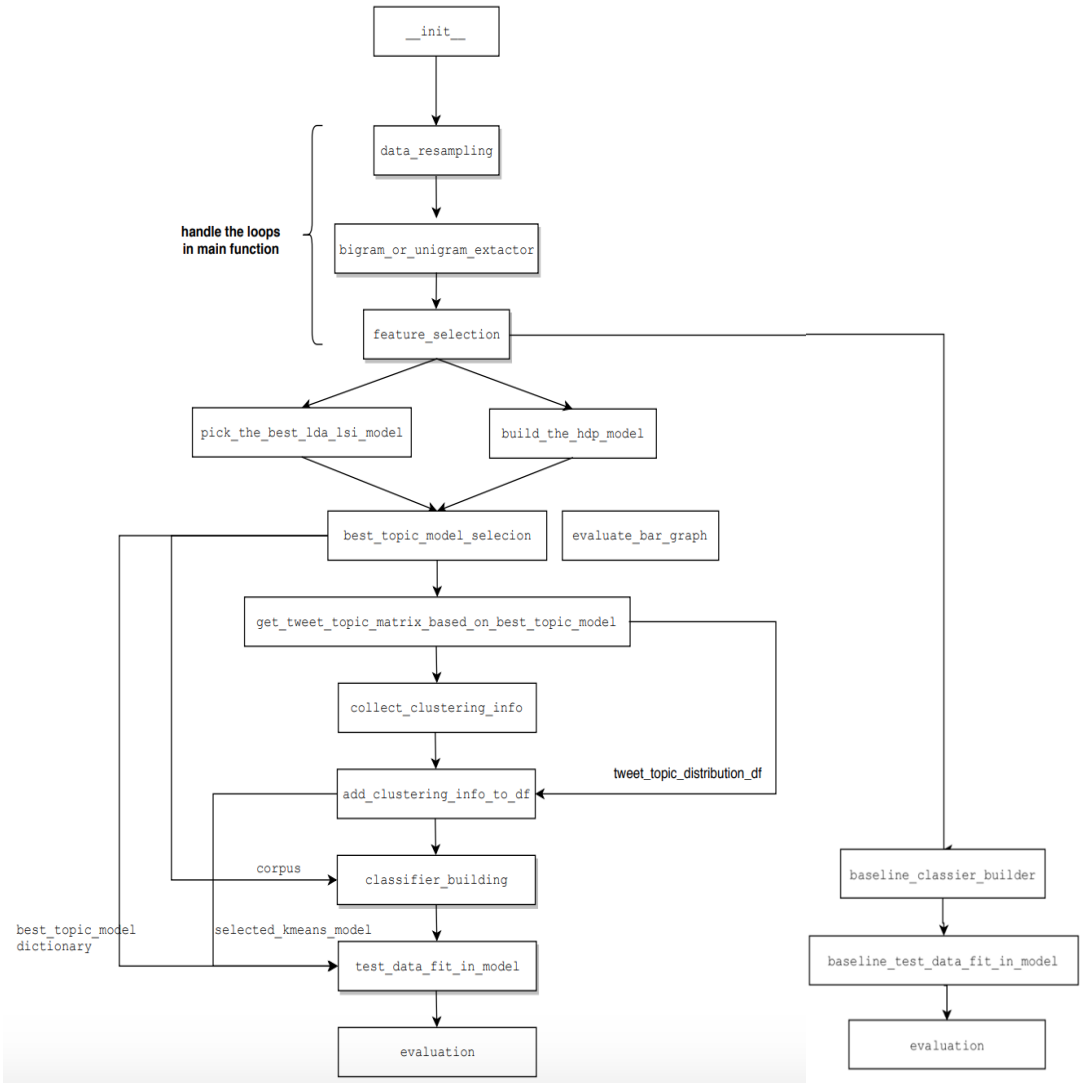


Illustration 10 Flow Chart of Function Level of Codes

The pseudocode in Table 5 shows the whole architecture of the system starts with data preparation. Also, a flow chart is shown in Illustration 10 about the function level of the codes.

In a code structure perspective, we can see that first of all, a data resampling function will handle the data imbalance problem by converting the imbalanced data to balanced data; After that, a feature extractor function is applied, and the outputs are N-gram features (unigram or unigram and bigram, depends on the parameter input by user);

This step is followed by feature selection by only keeping the top N features from the dataset; The next step is topic modeling part which includes LSI and LDA model building for several different topic numbers, and HDP model building which doesn't request the topic number parameter; The best topic model selection function is applied next to pick the best topic model/number of topic from all the topic models we built in the previous step; The following stage is getting the document-topic distribution of the selected model; K-means clustering algorithm is applied to the distribution to label the tweets into groups; Based on the different number of groups, corresponding amount of classifiers will be created on top of Logistic Regression and Random Forest algorithm; When finish building the models, test data will be fit in and evaluated; Apart from that, baseline models will also be trained as comparison methods.

4.6 Discussion

In this section, we want to discuss how our proposed topic-model based hybrid sentiment classification model is a novel model compared with the ones from previous works or from online tools.

The most import feature our proposed model has is the ability to choose the best topic model from a variety of topic models with different parameters settings. For topic modeling, the number of topics is normally required for models like LDA and LSI, which can be hard to determine without a deep understanding of the dataset. In previous works, some researchers made the number of topics as a fixed number (like topic number equals to 100 for all the runs applied by the model), and others apply HDP model which doesn't require to provide the topic number before modeling. While these approaches didn't solve the problem fundamentally. For the former approach, using a fixed topic number can bring subjectivity and bias, and most importantly, there is no guarantee that the model built with

the fix topic number would be the best model; For the latter approach, using HDP doesn't require the topic number while there is no way to verify the HDP model with self-determined topic numbers could outperform the LDA model with a fixed topic number. In our approach, we consider that deciding the topic number is just a way to find out the best topic model based on existing data but not the final goal, while the real goal is to find out the best topic model which can uncover the hidden semantic topics better than other models. Therefore, we build a pipeline in section 4.4 by employing three different topic models with different parameters setting, in order to find out the best model based on coherence scores. By doing this, there is a guarantee that the selected model for further clustering is the best one not only among the same topic model with different parameters, but also the best across the three topic models we use in the pipeline. We consider this is a novel method and one of our major contribution.

In addition, the whole process is running in an automated way by using coherence score for evaluation. Although there are several topic model evaluation methods, but none of the others can help us achieve our goal: Eye Balling models require human efforts and makes it hard to compare different models in a quantitative way; Perplexity could be used in an automated pipeline but it does not consider the coherence between the examined word and the topic; To align with our goal to make the whole pipeline as automated as possible and also consider the coherence between the word and the latent topic, coherence score is a good way to run evaluation among a series of topic models being built, and also easy to be integrate with the whole pipeline. It is also possible to compare the coherence score from models with different topic numbers to have a general idea of how the coherence scores change with the different topic numbers. While for the previous works we reviewed, perplexity is commonly used as a measurement for topic models. Therefore, the use of

coherence score to compare the performance among different topic models is also a novel method proposed in our study.

Chapter 5: Sentiment Classification Evaluation

In this section, we will report the sentiment classification result in terms of classification performance and efficiency analysis.

5.1 Classification Performance

In this section, we want to report our classification performance of our proposed solution in terms of accuracy, precision, recall, and F-measure which are the most commonly used performance metrics for sentiment analysis tasks. Moreover, we will compare the result with some previous work using the same dataset and compare the model design with some most recent works with different approaches.

5.1.1 Performance report on the proposed solution

The training dataset we used is collected by our own based on the method discussed in Chapter 3, with 290817 tweets with positive graphic emojis and 153324 tweets with negative graphic emojis. The full positive and negative emojis we used as filter can be found in Table 3.

We apply our proposed method and baseline model on Twitter in SemEval2013 Task 2-B as test dataset. There are three types of labels (positive, negative, and objective-OR-neutral) in the dataset. We only take the ones with positive or negative labels as our test dataset with 3120 positive instances and 3120 negative instances.

For topic modeling, we make the following filters: only words appear more than 5 times minimum and no more than 40% tweets would be used for topic model building. Other words would be discarded. For the model building part, we build k models for LDA and LSI where $k \in [3, 30]$.

For clustering, we set the minimum cluster number as 2 and the maximum cluster

number as 10. Then the final cluster number would be decided by human judgment based on Error Sum of Squares (SSE).

The feature representation is based on tf-idf and the feature selection is achieved by using Chi-square method which keeps the top 20000 features from n-gram matrix.

The classifiers used in the final classification task is Random Forest and Logistic Regression. We can also compare our proposed approach by applying pure Random Forest and Logistic Regression as baselines.

For performance evaluation, accuracy, precision, recall, and F-measure are reported. We choose these four metrics since they are the most popular ones for Twitter sentiment analysis tasks which can facilitate comparison among different studies. The formulas for the four metrics are:

$$\text{Accuracy} = \frac{\sum True\ Positive + \sum True\ Negative}{\sum True\ Positive + \sum True\ Negative + \sum False\ Positive + \sum False\ Negative}$$

$$\text{Precision} = \frac{\sum True\ Positive}{\sum Predicted\ Positive}$$

$$\text{Recall} = \frac{\sum True\ Positive}{\sum Actual\ Positive}$$

$$\text{F-measure} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

where true Positive refers to the correctly predicted positive values when the predicted class is positive, and the actual class is also positive; True Positive refers to the correctly predicted negative values when the predicted class is negative, and the actual class is also negative; False Positive refers to the wrongly predicted negative values when the predicted class is positive but the actual class is negative; False Negative refers to the wrongly predicted positive values when the predicted class is negative but the actual class is positive; Based on the formula, accuracy refers to the portion of correctly predicted

observation out of overall observations; Precision refers to the portion of correctly predicted positive observation out of overall predicted positive observations; Recall refers to the portion of correctly predicted positive observation out of overall actual positive observations; F-measure [108] is calculated by taking weighted average of Precision and Recall which is a balance between these two metrics. Compared with the other three metrics, F-measure is a more comprehensive metric when comparing among different models. Meanwhile, it is also commonly used for performance comparison among different sentiment analysis models when applying on the same test dataset.

The evaluation results are shown in Table 6.

Table 6 Classification Performance Evaluation Report

Model	Features	Accuracy	Precision	Recall	F-measure
Proposed method (LG)	unigram	65.64%	65.16%	67.22%	66.17%
	unigram+bigram	67.80%	67.13%	69.75%	68.42%
	unigram+bigram+lexion	74.62%	74.16%	75.58%	74.86%
	unigram+bigram+lexion+ negation	76.27%	75.88%	77.02%	76.45%
	unigram+bigram+lexion+pos	72.57%	71.73%	74.49%	73.08%
	unigram+bigram+lexion+ negation+pos	74.59%	73.69%	76.47%	75.06%
Proposed method (RF)	unigram	68.18%	67.62%	69.75%	68.67%
	unigram+bigram	70.99%	70.08%	73.25%	71.63%
	unigram+bigram+lexion	75.31%	74.53%	76.89%	75.69%
	unigram+bigram+lexion+ negation	78.26%	77.54%	79.56%	78.54%
	unigram+bigram+lexion+pos	74.97%	74.50%	75.93%	75.20%
	unigram+bigram+lexion+ negation+pos	76.47%	75.20%	79.01%	77.06%
Baseline LG	unigram+bigram+lexion+ negation	59.36%	59.13%	60.63%	59.87%
Baseline RF	unigram+bigram+lexion+ negation	61.83%	61.63%	62.69%	62.16%

N.Malandrakis et al. [28]	POS, ngram	-	-	-	77.65%

Table 6 shows how the model performs when adding more features to it. It is clear that all the results driven by our proposed model are performing better than the two baseline counterparts.

For our proposed methods with different base classifiers, we all start with using unigram features and gradually add more features into the model. The best results are achieved by our proposed approach with Random Forest. For this model, it is clear that using unigram leads to acceptable results but not much more significant than the Random Forest baseline. After adding bigram features, the performance increases slightly but not too significant. Adding lexicon features have a positive impact on the performance which boost both the precision and recall increasing and lead to an around 4% improvement on F-measure. The best performance is achieved after adding negation features into the model based on the previous features with a 78.26 accuracy, 77.54 precision, 79.56 recall, and 78.54 F-measure. We also try to add part-of-speech into our model, but it ends up with decreasing our best performance based on the previous combination.

Generally, the other proposed approach with Logistic Regression doesn't perform as good as the Random Forest counterpart but outperform the two baseline algorithms. A significant improvement could be seen when comparing with the Logistic Regression baseline which indicates our proposed model does help to capture some semantic level information to help separate the tweets and boost the performance.

5.1.2 Performance comparison with other works

After comparing the usage of different features and different supervised learning classifiers, we also want to compare our proposed approach with previous works. Since we

are doing a binary classification task in our study based on SemEval-2013 task 2-B, which provide 3 classes (positive, negative, neutral) in training and test dataset, there are not many works available for comparison. While [28] reports the results of its model both on 3-way (positive, negative or neutral) and binary classification (positive or negative) based on SemEval-2013 task 2-B as test dataset, so that it becomes a potential work we can compare our result with. It would be better to compare the classification performance if it is possible to utilize the same training dataset and test dataset, but only different sentiment classification models on top of them. While since study [28] didn't describe all the details of their proposed system (there are only 4 pages without the references including the introduction, Experimental procedure, Results, and Conclusion), it is hard for use to re-implement their system and run a side by side comparison. Therefore, our comparison is applied based on different training datasets, different features representations, different model design, but the same test dataset. The best F-measure (78.54) of our model is achieved by using N-gram features, lexicon features, and negation features driven by Random Forest algorithm which outperforms their binary classification average F-measure (77.65) for unconstrained conditions.

We also want to compare our proposed solution with some most recent studies. Since different datasets are used for these researches and ours, direct comparison based on performance is not reasonable. Therefore, we will analyze the difference between our research and theirs in terms of data collection, overall approach, system performance.
Since hybrid approaches are drawing more and more attention so that we want to start with comparing our approach with the hybrid approaches that are published in 2019.

[74] is a novel hybrid approach that is the first one proposes Sentiment Diffusion Patterns which can help to increase the performance of existing models especially for the

cases a lot of retweets could be observed in the training corpus. While although with an F-measure as 80.91, this approach tends to work well only in the following conditions: 1) There is a significant number of accessible retweets in the corpus; 2) The retweets must be clean and meaningful; 3) The retweet flow is accessible when collecting the data. While in reality, the first two conditions are not guaranteed for streaming data from tweets which is the foundation that this feature could add benefit for any existing algorithm. Even if the corpus does include a lot of retweets, most of them tend not to have "sentiment reversal" but only a simple "RT" means retweet. Also, from the annotation work that is done in the study, it is clear that how much human involved works need to be done to build such a training corpus which will block the model to be automated. Last but not least, condition three might be related to some privacy issue and need to be carefully treated. As a system, our proposed approach doesn't require conditions, is agile to the new coming data, and easy to configure in terms of data collection and data storage.

[72] is another most recent study using hybrid method for sentiment analysis. The solution they proposed could boost the performance by using several features including domain-specific ontology features, entity features, lexicon features, conceptual feature. While this approach is more focus on domain-specific problems while our system is more flexible to constantly change. For example, our corpus could be updated on a monthly basis without extra efforts and the whole pipeline could run the exact same way as before. While their approach requires delicate ontology design which needs to be maintained by people (for domain-specific tasks maybe it doesn't require a high frequency for sure), and the conceptual features are built based on Alchemy API tool, which means the semantic concept is fixed (or the changes will be entirely up to the API). Overall, our system contains more flexibility and consistency.

[41] is another hybrid approach which is mostly driven by lexicon resources. To be more specific, the slang classifier is powered by a rule-based slang dictionary collected from four online resources; The emoji classifier is built by organizing five existing lists into a single dictionary; The domain-specific classifier is utilized to correct the polarity score for domain-specific words by using frequency-based probability (Smeureanu &Bucur, 2012). Compared with their work, we create a more state-of-the-art emoji dictionary by including the graphic emoji rather than only the string emojis which could capture more emotion-related features in the tweets; Meanwhile, all their work is based on only term level but miss the semantic meanings that could be expressed by phrases. In contrast, we consider bigram as one of the features which utilizes more information from the order of the words.

[50] is a recent paper that utilize deep learning (CNN) for Twitter sentiment analysis. A similar evaluation is applied on SemEval 2014 task 9 dataset which is the same with our SemEval 2014 task 9 dataset task 2 dataset and the best F-measure for this dataset is 81.99. While compared with our best F-measure 78.54, This result is achieved by pre-train GloVe model using a 20 billion twitter corpus including 200-dimensional word vectors (and probably with much bigger computational resources). Considering the time and resources, our approach will be a more proper architecture for small to medium size business use cases.

5.2 Efficiency Analysis

In this section, we will discuss the efficiency of our system. The time complexity of our model, running time, CPU time, and memory usage will be analyzed in the following sections. Our device is MacBook Pro with 2.2GHz Intel Core i7 processor, memory 32 GB 2400 MHz DDR4. All the efficiency data reported are based on runs on this device.

5.2.1 Complexity analysis

In this section, we want to discuss the complexity of our sentiment classification model. Our model contains several different components for different functionalities. The major parts that are time-consuming includes topic model building, clustering, feature selection, and classifier building. We will go into the complexity details of each above-mentioned components and go back to the overall complexity.

5.2.1.1 Time complexity of topic modeling

As introduced in the previous chapter, topic modeling is the first step that we apply to our clean data after feature selection to uncover the latent topic of the training dataset. Overall, three topic models are used in this study: LSI, LDA, and HDP. For the implementation, we utilize the Gensim package in python for topic model building.

For LSI model, it is implemented based on paper [109] which demonstrates the power of randomization for performing low-rank matrix approximation that outperforms its traditional competitors according to accuracy, speed, and robustness. Based on the paper, we can see that the time complexity for building an LSI model is O(mnlog(k)) floating-point operations with a m × n matrix and a target rank k.

The popular inference methods for LDA include Gibbs sampling and variational inference. Gibbs sampling samples from the conditional distribution of all other variables and is usually considered easy to implement; While for variational inference, each factor is set to the exponentiated log of the conditional and is regarded easier to parallelize. In Gensim, LDA is implemented based on [110] which uses variational inference for scalability purpose.

For M as the document amount, γ as the Dirichlet distribution of variational

inference, ϕ as the multinomial distribution from LDA model, corpus as C, L as an objective function, λ as expectation of variational parameters for the solution to latent variables, the general idea of the variational inference for LDA is shown in Table 7.

Table 7 Variational Inference for LDA

Algorithm: Variational Inference for LDA

Initialize the variational parameters randomly;
For t=0 to ∞ do:
  For I = 1 to M do:
  update γ and ϕ;
  update β_c;
  if L < 0.0001:
  stop;
  else:
  continue;
return λ;

For each iteration, the time complexity equals to $O(M(T_\gamma+T_\phi) +T_\beta+ T_L)$ and it is shown that Online variational inference for LDA converges much more faster than Batch variational Bayes for LDA method for large dataset.

HDP model is implemented on top of [111] which also utilize variational inference for model building. With the gradient of the corpus-level parameters for top-level sticks and the topic: λ, u, v, and document-level variational parameters: the parameters to the per-document stick a_j, the parameters to the per-document stick b_j, the parameters to the per word topic indicators ϕ_j, and the parameters to the per document topic indices ζ_j; an appropriate learning rate p_{t_0}, the general idea of the variational inference for HDP is shown in Table 8.

Table 8 Variational Inference for HDP

Algorithm: Variational Inference for HDP

Initialize the variational parameters λ, u, v randomly;
For t=0 to ∞ do:
  For I = 1 to M do:
  update a_j, b_j, ϕ_j j and ζ_j;
  update ∂λ(j), ∂u(j) and ∂v(j);
  update p_{t_0};
  update λ, u and v;
  if $\sum_{t_0=1}^{\infty} p_{t_0} = \infty$, $\sum_{t_0=1}^{\infty} {p_{t_0}}^2 < \infty$:
  stop
  else:
  continue;
return λ, u, v;

Based on the pseudocode above, the time complexity of each iteration for variational inference for HDP algorithm is $O(M(T_d+T_t+T_p+T_L))$ with T_d as the time consumed for document-level variational parameters and T_t as the topic-level variational parameters, T_p as time for updating the learning rate, and T_L as the time for calculating that whether or not converge standard is met.

5.2.1.2 Time complexity of clustering

The second time-consuming component of our system is clustering based on the document topic distribution which is the output of topic modeling part. In this part, K-means is used as our clustering method and the implementation in sklearn[39] is based on

[39] https://scikit-learn.org/stable/

paper [112]. With where K is the number of clusters, the time complexity of this K-means implementation is O(logK) which is improved in terms of speed and the accuracy based on the original K-means algorithm.

5.2.1.3 Time complexity of feature selection

Chi-square[92] method is applied for the feature selection part of our study. For the corpus with M feature and N classes, the time complexity of Chi-square test is O(MN). In our implementation, there is another step to rank the features based on their Chi-square scores and take only the top N ones. While the time of that part is not a major part for the time consumed for feature selection part.

5.2.1.4 Time complexity of classifier building

In the classifier building part, two classifiers are used to build models based on our processed dataset in previous steps: Logistic Regression and Random Forest. We will discuss the time complexity for both of them.

Logistic Regression is a simple and powerful tool for classification tasks. For building a model based on the training dataset, Gradient Descent technique is usually used to search for the most optimized curve by estimating the error between current simulation and the observations. We can see that the sklearn module also achieve the implementation based on it. Based on Gradient Descent, the time complexity for building a binary logistic regression model is O(fs) for each iteration, with f as the number of the features and s as the number of instances in the corpus.

Random Forest is an ensemble model based on decision tree. Therefore, the time complexity of Random Forest algorithm is based on the complexity of decision tree multiply the number of trees being built overall. For building a single CART tree by using

n observations and m features, the time complexity is O($mn\log n$). If the Random Forest consists of K trees, the final complexity is O(K($mn\log n$)).

The overall time complexity of our system could be regarded as a linear combination of the time complexities from all the components. So, with the above-mentioned parameters, the system complexity is:

Overall complexity = O(mnlog(k)) + O(R_{LDA} M($T_\gamma + T_\phi$) + T_β + T_L) + O(R_{HDP}M(T_d+T_t+T_p+T_L)) + O(logK) + O(MN) + O(R_{LR}fs) + O(K_{tree}($mn\log n$))

where R means the round required for converge.

The actual running time for each part also depends on other constant. For example, the K-means result can determine the number of Logistic Regression and Random Forest classifiers would be built later. Therefore, the overall complexity only shows the increasing trend for each part.

5.2.2 Program running time

Program running time refers to the actual running time for a specific program. In Table 9, we list several major components that take significant time or are essential parts in our program with a different combination of features so that we can have a general idea how long it would take for each part to run.

Table 9 Program Running Time Report

Running Time (seconds)	**unigram**	**unigram+ bigram**	**unigram+ bigram+ lexion**	**unigram+ bigram+ lexion+ negation**	**unigram+ bigram+ lexion+ pos**	**unigram+ bigram+ lexion+ negation+ pos**
Overall	2541.79	2567.13	2458.49	2420.86	2362.75	2434.98
LDA model	1294.11	1305.05	1246.52	1245.58	1252.29	1306.23

building						
LSI model building	302.93	316.68	306.45	301.95	292.04	294.00
HDP model building	206.61	214.95	209.91	206.96	202.26	208.69
K-means clustering	30.06	21.80	17.60	38.00	18.08	21.83
Feature extraction	15.07	15.27	15.69	15.10	15.10	15.48
Feature selection	405.19	415.11	456.21	408.55	401.27	402.30
Classifier building	85.37	88.71	81.30	76.98	72.60	77.13
Test data fit in	3.40	3.24	3.81	3.86	3.67	3.66
Baseline classifier building	94.53	83.44	120.99	123.87	105.44	105.66
others	104.53	102.88	105.52	115.49	117.26	130.17

The data from the table shows the running time of the runs with different feature combinations when applying Logistic Regression. The overall running times vary between 2363.75s to 2567.13s. The parts that taking significantly longer time to run compared with other ones are topic model building and Feature selection. LDA model building, LSI model building, HDP model building take 1294.11s, 302.93s, and 206.61s respectively which accounts for more than 50% of the overall running time of the program. Feature selection which takes more than 400s to be finished also contribute a lot to the entire running time.

Since our feature selection is applied to the N-gram features, so adding lexicon, negation, and POS features don't improve the running time significantly.

5.2.3 CPU time

Compared with program running time or elapsed real time, CPU time stands for the amount of time for which a CPU was used for processing instructions of a computer program or operating system[40]. Specifically, CPU time could be broken down into CPU user time and CPU system time. CPU user time is the amount of time the processors spent on the particular program you run while CPU system time operating system is the amount of time the processors spent on the functions connected to that specific program. Table 10 shows the CPU time and the breakdowns of the two subcategories for different runs.

Table 10 Program CPU Time Report

CPU Time (seconds)	**unigram**	**unigra+ bigram**	**unigra+ bigram+ lexion**	**unigram+ bigram+ lexion+ negation**	**unigra+ bigram+ lexion+ pos**	**unigram+ bigram+ lexion+ negation+ pos**
Overall	5302.09	5343.12	5424.41	5462.88	5313.48	5487.86
CPU user time	4997.37	5027.21	4982.70	5004.78	4877.29	5020.98
CPU system time	304.72	315.91	441.72	458.10	436.19	466.88

It is clear that although there is no significant difference in the program running times among different runs, we can see the CPU times are increasing when we add more features into the model. Furthermore, the increase of overall CPU time is not due to the

[40] https://en.wikipedia.org/wiki/CPU_time

CPU user time which tends to be stable across these runs but because of the CPU system time which doing more functioning connections for our program.

5.2.4 Memory usage

Memory usage refers to the volume of memory is used by your program. With the help of memory-profiler[41], we can monitor the memory usage over time for our program. The two figures below show the memory usages for our system when using Logistic Regression and Random Forest as the base classifiers.

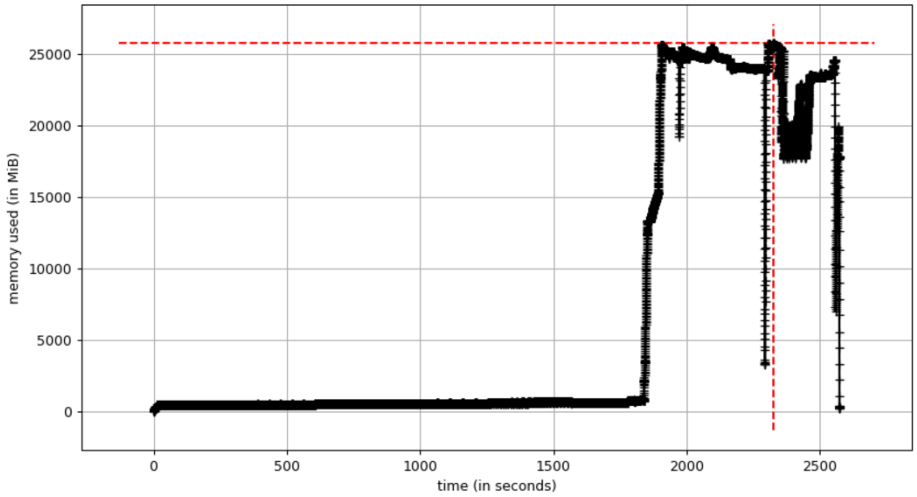


Illustration 11 Memory Usage for Logistic Regression

The Memory Usage of Logistic Regression is shown in Illustration 11. For the program that using Logistic Regression as the base classifier, the memory usage is quite few for the topic modeling parts while it starts to grow when the feature selection starts and followed by the classifier building and training data fit into the process. The memory usage peaks at more than 25000M for that moment. The memory-intensive process lasts for around 700-800 seconds.

[41] https://pypi.org/project/memory-profiler/

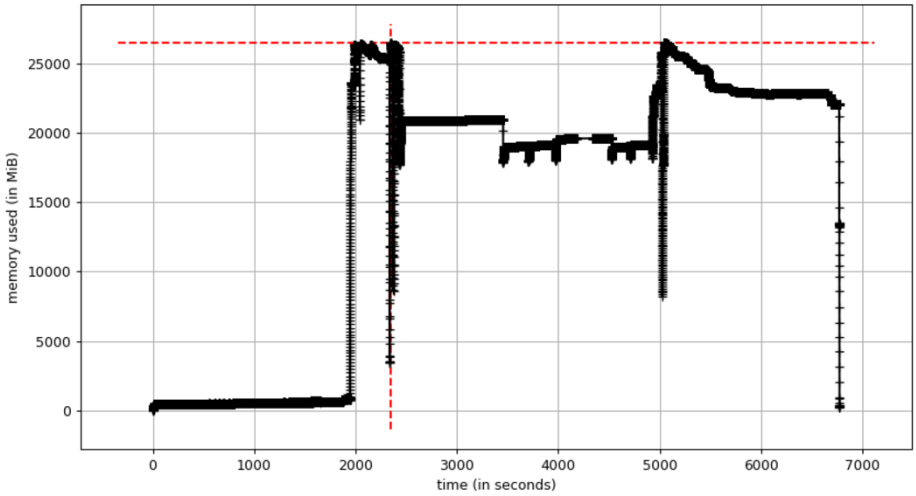


Illustration 12 Memory Usage for Random Forest

The Memory Usage of Logistic Regression is shown in Illustration 12. For the program that using Random Forest as the base classifier, the memory usage is also quite insignificant for the topic modeling parts. After topic modeling, it starts to grow when the feature selection starts and followed by the classifier building and training data fit into the process as well. The memory usage peaks at more than 26000M for that moment and more than 4000s the memory usage is more than 20000M. We can see that running Random Forest algorithm is much more memory intensive than Logistic Regression.

5.2.5 Recommendation

Across the whole program, we can see that topic modeling requires relative high CPU time while but low memory. In contrast, the feature selection and classifier building parts require more memory usage than the topic modeling, especially when it comes to Random Forest model. Based on these observations, we can adjust our topic modeling, feature selection, and classifier building strategy based on the current device settings.

5.3 Analysis and discussion

We want to take a further look into the model to see how we get this result and also how could people tune this model for their own cases. One of the essential parts is the topic

modeling part since it determines how we separate our training data to build the classifiers later.

We apply three topic models in our study: LDA, LSI, and HDP. By running with the parameters for our best run, Illustration 13 illustrates how the coherence score for LDA model changes over the increase of the number of topics k between [3, 30]. It is clear that when the number of topics varies from [6, 10], we get the best coherence especially when it comes to k = 8 or k = 10; When k ∈ [11, 19], the scores are slightly lower than before; While when k is greater than 19, the coherence scores witness a downward trend till the very end.

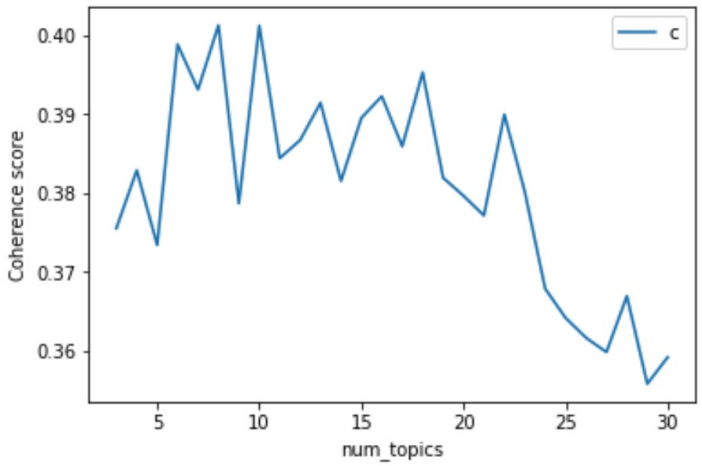


Illustration 13 Coherence Scores for LDA

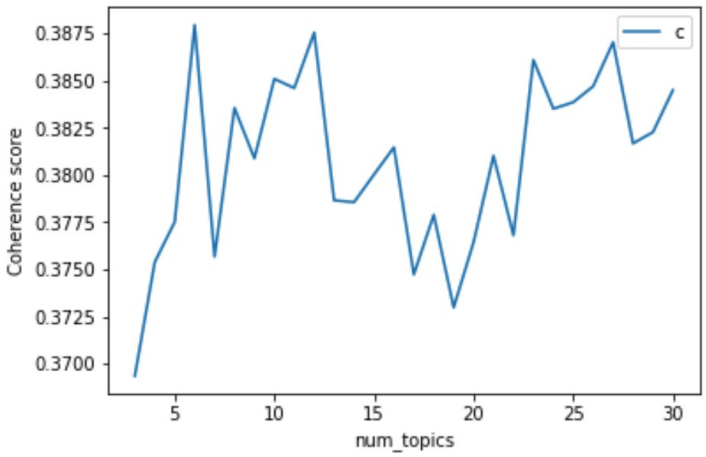


Illustration 14 Coherence Scores for LSI

Illustration 14 shows the coherence score graph for LSI model. We can see that the coherence scores are quite high when k ∈ [6, 13]. After that, there is a significant drop between [14, 22]. Interestingly, when k is greater than 23, the coherence score going up again. Overall it doesn't show a stable trend like what we saw for LDA models.

For HDP model, it will select the optimized number of topics which is 20 in our case.

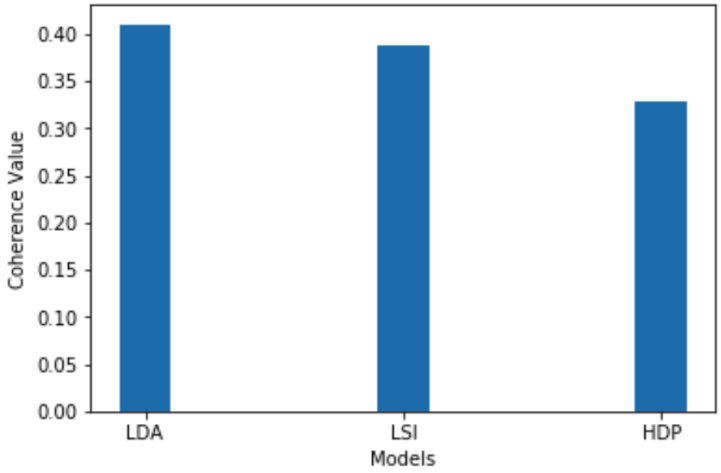


Illustration 15 Coherence Scores Comparison for LDA, LSI, LDP

A comparison of the best coherence scores for the three models is shown in Illustration 15. We can see that LDA model (0.41) outperform the LSI model (0.39) and HDP model (0.33) in this run. It is aligned with the experience of people that LDA always achieve better results in practice.

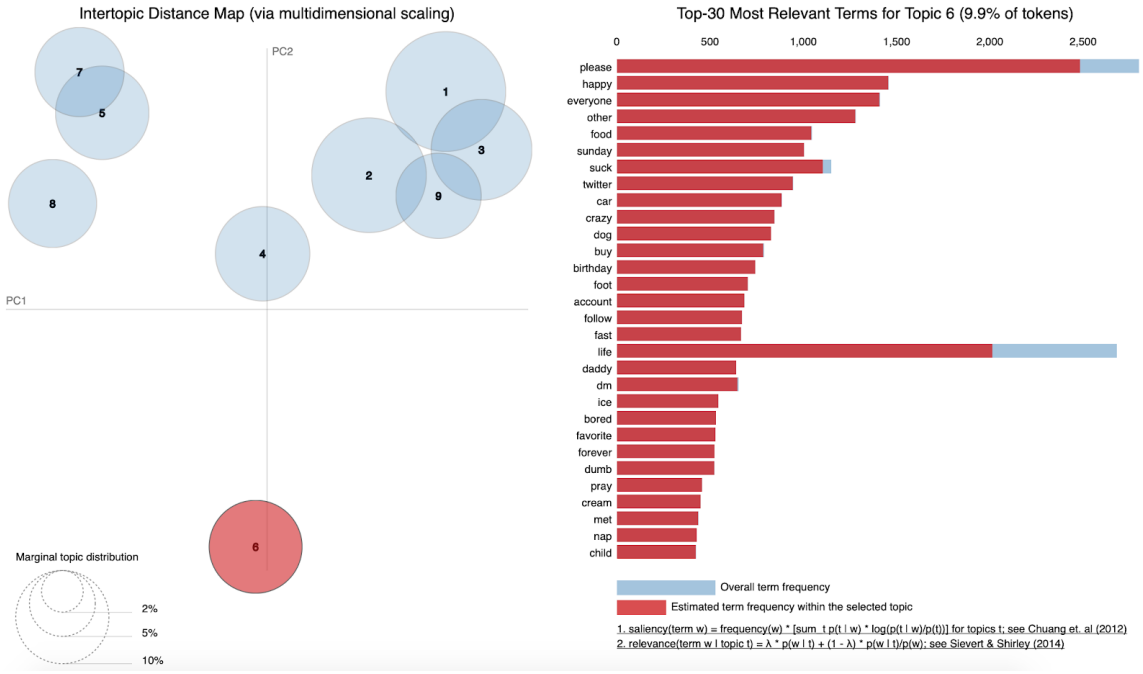


Illustration 16 Topic Illustration for LDA – Topic 6

We also want to illustrate the topics we build for our model. With the help of LDAvis [100], we can map the topics into 2D dimension by using Principal Components Analysis. We can see that this topic 6, which is shown in Illustration 16, is a topic with mostly positive words like ‘happy’, ‘birthday’, ‘favorite’, ‘sunday’, etc. Also, this topic is significantly having distance with other topics.

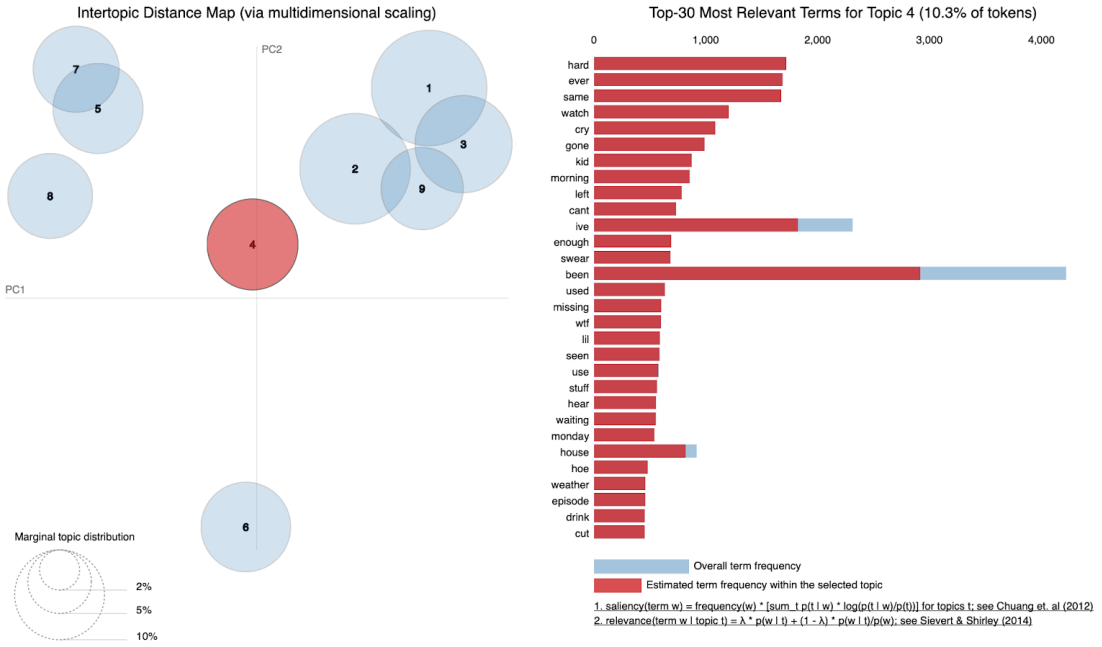


Illustration 17 Topic Illustration for LDA – Topic 4

In contrast, topic 4, which is shown in Illustration 17, tend to be more negative compared with topic 6 with words like ‘hard’, ‘cry’, ‘gone’, ‘swear’, ‘missing’, etc. becoming the most relevant words. Topic 4 is also not overlapped with other topics and interestingly show a different sentiment compared with topic 6.

While other topics like topic 1,2,3,9 which have overlaps among them may not have difference as different as the above two topics. A whole list of all topics with their corresponding probabilities in Illustration 18. We can see that topic modeling on tweets is more challenging than other corpora.

	topic_0_words	topic_0_probs	topic_1_words	topic_1_probs	topic_2_words	topic_2_probs	topic_3_words	topic_3_probs	topic_4_words	topic_4_probs	topic_5_words	topic_5_probs	topic_5_words	topic_5_probs	topic_6_words	topic_6_probs	topic_7_words	topic_7_probs	topic_8_words	topic_8_probs
0	i	0.144158	im	0.072757	is	0.046376	rt	0.052791	i	0.078481	i	0.064741	i	0.064741	are	0.062169	i	0.180045	is	0.076473
1	not	0.140391	is	0.053883	been	0.027791	please	0.024400	love	0.054127	im	0.037057	im	0.037057	is	0.052343	need	0.040180	so	0.042801
2	do	0.062573	so	0.040864	time	0.022439	day	0.022926	rt	0.030896	got	0.027173	got	0.027173	rt	0.025996	want	0.033548	im	0.030851
3	know	0.022563	are	0.018976	have	0.022363	life	0.019774	baby	0.029462	wa	0.024910	wa	0.024910	friend	0.024022	just	0.031661	damn	0.028779
4	have	0.022026	rt	0.017746	rt	0.020241	is	0.017054	fuck	0.023940	so	0.020926	so	0.020926	man	0.023300	wan	0.027318	gon	0.027047
5	did	0.020052	feeling	0.017566	i	0.018610	happy	0.014295	bitch	0.022890	back	0.019831	back	0.019831	being	0.022160	na	0.021525	look	0.025477
6	miss	0.017730	yes	0.014194	ive	0.017369	everyone	0.013823	said	0.017943	be	0.019735	be	0.019735	so	0.017292	rt	0.020771	rt	0.022776
7	shit	0.015886	about	0.011634	hard	0.016369	u	0.013593	too	0.015610	had	0.018809	had	0.018809	omg	0.017243	wa	0.019128	are	0.018994
8	im	0.015060	much	0.011589	ever	0.016056	other	0.012572	nigga	0.014928	sad	0.013773	sad	0.013773	cute	0.016524	wrestlemania	0.017180	be	0.015103
9	u	0.015038	be	0.011063	same	0.015945	let	0.012534	much	0.013773	get	0.013161	get	0.013161	having	0.013364	really	0.014809	na	0.014086
10	is	0.014916	sorry	0.010778	good	0.014207	amp	0.011373	tho	0.013708	is	0.012968	is	0.012968	never	0.013063	stop	0.013233	heart	0.013914
11	get	0.014421	game	0.010736	now	0.012421	suck	0.010857	school	0.013363	have	0.012880	have	0.012880	put	0.012316	be	0.013043	someone	0.013859
12	am	0.013734	thank	0.010631	be	0.011914	food	0.010240	is	0.012893	day	0.012602	day	0.012602	weekend	0.011647	feel	0.011814	rn	0.013613
13	just	0.012864	soo	0.010366	watch	0.011473	sunday	0.009855	talk	0.012880	rt	0.012284	rt	0.012284	little	0.010647	go	0.011744	oh	0.013048
14	hate	0.012817	doing	0.009747	never	0.010853	come	0.009629	face	0.010000	wish	0.011992	wish	0.011992	w	0.010433	is	0.010775	there	0.012522
15	even	0.012292	stay	0.009055	cry	0.010339	twitter	0.009271	somebody	0.009776	sleep	0.011405	sleep	0.011405	head	0.010315	hair	0.010390	cause	0.012505
16	think	0.010985	really	0.009028	gone	0.009411	go	0.009049	mom	0.009281	go	0.011242	go	0.011242	such	0.010093	literally	0.010083	just	0.012007
17	be	0.010034	yeah	0.008631	ha	0.009154	only	0.008754	id	0.009027	just	0.011033	just	0.011033	okay	0.010058	am	0.009972	god	0.011763
18	rt	0.009658	u	0.008184	best	0.009023	car	0.008677	missed	0.008612	again	0.010818	again	0.010818	too	0.009233	thought	0.008958	hit	0.011420
19	want	0.009452	anymore	0.008130	thing	0.008418	get	0.008465	make	0.008135	girl	0.010628	girl	0.010628	nice	0.008602	get	0.007585	idk	0.011374
20	really	0.009288	mad	0.008126	kid	0.008312	crazy	0.008294	honestly	0.008042	ill	0.010519	ill	0.010519	away	0.008239	whole	0.007047	fucking	0.010878
21	wait	0.008858	just	0.007894	amp	0.008306	dog	0.008122	got	0.007803	last	0.010502	last	0.010502	took	0.007839	see	0.006681	wow	0.010060
22	so	0.007788	bad	0.007527	morning	0.008146	make	0.007813	tf	0.007587	tomorrow	0.010103	tomorrow	0.010103	name	0.007488	looking	0.006543	good	0.009958
23	doe	0.007576	very	0.007141	house	0.007802	buy	0.007718	pain	0.007397	bad	0.009963	bad	0.009963	black	0.007464	song	0.006491	sick	0.009850
24	hurt	0.007409	making	0.006883	left	0.007462	pls	0.007686	girl	0.007079	year	0.009377	year	0.009377	alone	0.007324	actually	0.005851	body	0.008692
25	feel	0.006984	hell	0.006802	im	0.007215	big	0.007579	night	0.007019	tired	0.009268	tired	0.009268	leave	0.007138	try	0.005433	here	0.008642
26	wa	0.006563	trying	0.006605	still	0.007012	birthday	0.007290	end	0.006891	week	0.009191	week	0.009191	mood	0.007040	home	0.005271	pretty	0.007952
27	already	0.005829	worst	0.006546	cant	0.006980	new	0.006938	bed	0.006373	ta	0.008600	ta	0.008600	si	0.006661	call	0.004857	really	0.007845
28	see	0.005735	season	0.006183	long	0.006717	foot	0.006892	just	0.006265	now	0.008226	now	0.008226	job	0.006590	more	0.004780	wrong	0.007782
29	people	0.005715	real	0.006178	enough	0.006582	help	0.006775	nobody	0.006155	work	0.007793	work	0.007793	family	0.006458	never	0.004733	wanted	0.007704

Illustration 18 Topic Word Distribution for LDA

As a social network, people tend to talk more about casual topics by using informal languages instead of serious topics and using formal languages. This does bring more changes to data analysis. Another assumption is maybe the major part of tweets sample is random topics while a small portion of them is a more organized and better material for topic modeling just like what we do for news. Therefore, finding out a way to separate the formal part and informal part might boost the performance further based on what we get today.

Chapter 6: A Use Case and Pipeline Discussion

After introducing the proposed data collection and sentiment classification methods, an important following step is how to use them to solve real problems. In this Chapter, we will start with a use case to illustrate how to apply our proposed pipeline on a real business question and followed by analysis and discussion of the pipeline.

6.1 A Use Case Study

As we mentioned in the Introduction chapter, Twitter sentiment analysis has a lot of potentials to drive commercial products, and we want to use a use case to show how to apply our proposed pipeline to drive commercial applications. Assuming that we are employees that are working in marketing or public relation department of an Internet company and want to know what people are posting towards our brand, and most importantly, what are their sentiment towards our brand. This use case is built based on this assumed scenario.

So far, we have introduced the details about our proposed data collection and sentiment classification methods. While in a real use case, there is something else that needs to be considered too, especially if the methods are used to drive a business product. The first thing on the list is data freshness, which includes the freshness of training dataset, sentiment classification models, and front-end dashboard. The reason why we need to care about data freshness is that applying sentiment analysis in a business product is not an ad-hoc task, but a long-term request that needs to be run periodically. Therefore, it is important to provide the stakeholders with the most updated results for their decision-making process by ensuring the freshness of data and model and dashboard in the pipeline. Another consideration is model performance evaluation. Since we need to ensure the data freshness,

it is expected that a variety of sentiment classification models will be built based on different groups of training data with different freshness. When data freshness is satisfied, there is no guarantee that the latest model is the best model among all the models being built. To solve this problem, a model evaluation step is necessary every time a new model is built to make sure that the selected model to be applied on the test data is the best model we can get. Due to the above two considerations, we will provide details about how to design the data processing steps.

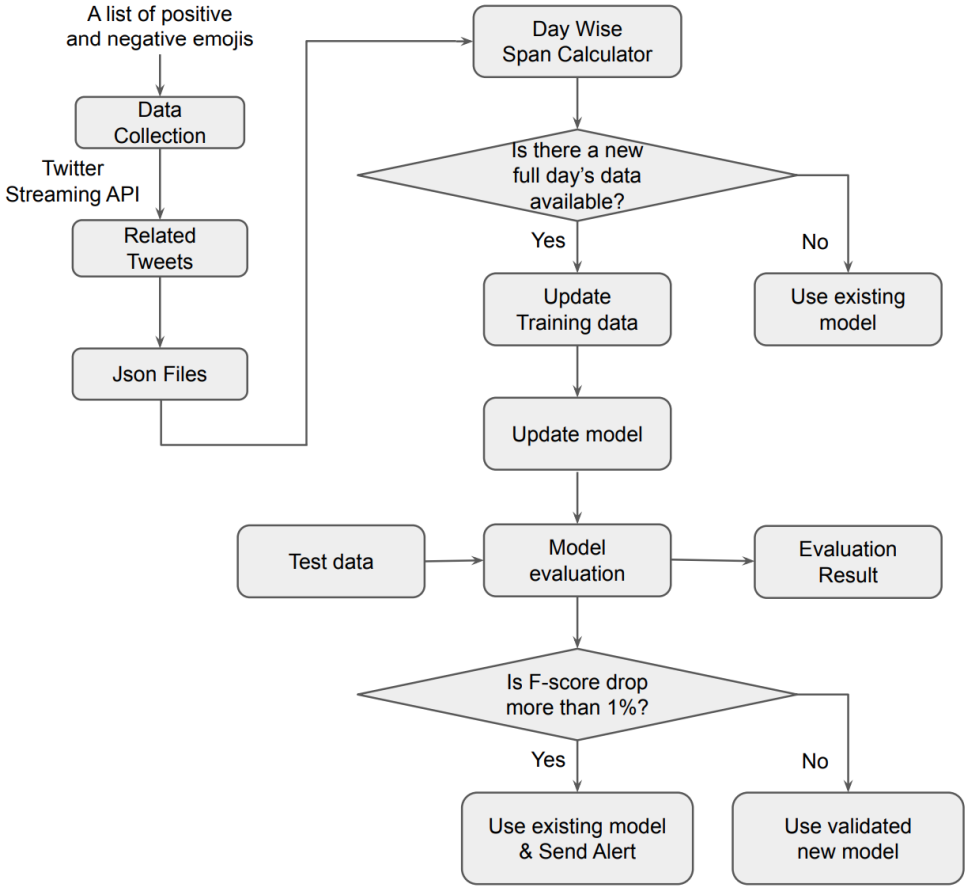


Illustration 19 Data Processing Steps for Sentiment Classification Model Updating and Evaluation in Use Case

Firstly, we can start with the data processing steps for sentiment classification model updating and evaluation based on our proposed data collection method in Chapter 3

and proposed sentiment classification method in Chapter 4. As it is shown in Illustration 19, this part starts with data collection which aims at collecting most recent tweets which will be used as the training dataset later which are stored as JSON files once collected. After that, we plan to start the model building step in a daily basis once yesterday's data becomes complete and available. Therefore, a day wise span calculator, which is a function to check the completeness of the data for the past day, will be applied first. If there is a new full day's data available, then the newly collected tweets will be added into the training dataset to replace the data from the oldest day in the training dataset; Otherwise, there will be no update on the training dataset and the existing model will be used. There will be 7 days data minimum being used in the training dataset.

Once the training dataset is updated, the model will be updated too as discussed in Chapter 4. When the newest model is ready, naturally we need to know whether it performs better, same, or worse than the current model. We surely expect it to be better, but it is necessary to have a step for model evaluation, which can also help us to trace the performance trend over time. As it is shown in Illustration 19, the new model will be applied on the labeled test data for evaluation purpose. If the F-measure of the newest model is lower than the current model, then it means the new model is no better than the current model. In this case, the current model will still be used, and an alert will be sent to the developer (developer can check if there is any noise exist in the newly collected data or other checks if necessary); In the other cases, the newest model will be used for the following classification tasks. By doing this, the model being used will always be the one with the best performance.

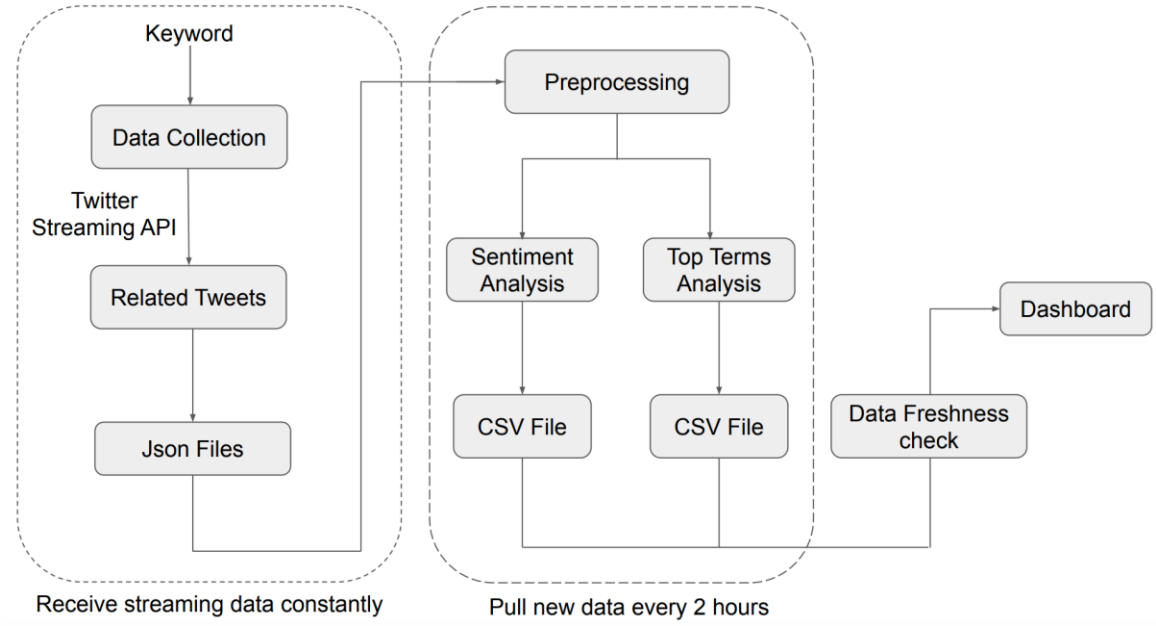


Illustration 20 Data Processing Steps for Dashboard Refreshing in Use Case

After talking about the data processing steps for updating and evaluating sentiment classification model, we want to discuss how to utilize the established model to monitor the sentiment orientation and trend for brand-related tweets based on the provided brand keyword. Illustration 20 shows the data preprocessing steps about receiving the tweets that we analyze related to our brand, and how to refresh the front-end dashboard in an hourly basis. To monitor the sentiment of brand-related tweets, we need to acquire the relevant tweets as a beginning. In this part, a keyword will be asked about the brand or topic that is expected to be monitored. After the keyword is provided to the system, the system will start to send requests to Twitter Streaming API to collect these tweets and store them as JSON files. In order to trace the sentiment trend over time, once the keyword is provided, the requests will be kept sending to Twitter Streaming API unless the user chooses to stop. By doing this, the system can collect the related tweets from the time the input is provided to the ongoing time to get a complete history of this keyword. The assumption behind this design is the company or institution which wants to use this tool will want to monitor the

relevant information about their own brand or topic, so that the needs won't change frequently.

Compared with data collection which will be processed continuously, the data analysis part will be triggered in an hourly basis. Currently, it is designed to pull the data every two hours, but the user can change the frequency based on their preference. The pulled data will be preprocessed first and then fit into different functions for sentiment analysis and top term analysis (shown in Illustration 21, Illustration 22, and Illustration 23). The result will be saved as csv files which is easy to be used as inputs of the visualization tool (Tableau) in the following step. As a following step, a data freshness function is constantly running to check whether there are new csv files generated from the previous step based on the timestamp. If there is any, the dashboard will switch to the most recent files, and the front-end dashboard will be updated too.

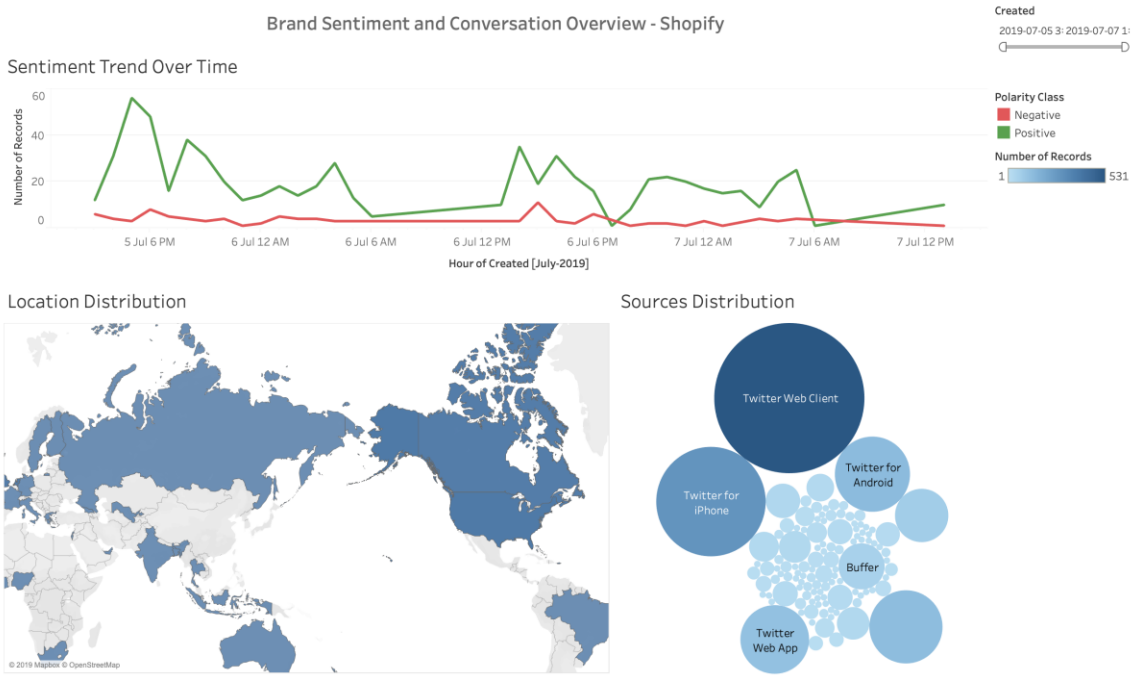


Illustration 21 Brand Sentiment and Conversation Overview Dashboard

In this use case study, three dashboards are made to illustrate the brand sentiment

and conversation overview, overall term discussion, and hourly top tweets. The keyword provided to update the dashboards in this case is “Shopify”, which is a Canadian e-commerce platform that helps merchants selling products. Imagine we are the marketing people in Shopify and want to know what people are talking about in Twitter about Shopify in terms of experience, news, complaints, and anything else, especially the sentiment towards these comments.

First of all, the overall sentiment trend could be read from the first dashboard: Brand Sentiment and Conversation Overview – Shopify, which is shown in Illustration 21. This dashboard provides an overview of sentiment trend against Shopify and also the location and source distribution. It is clear that the tweets with positive sentiments have a dominating amount compared with the ones with negative sentiments. There is a spike for positive sentiments on 5th July, 5:00 pm and an increase on negative sentiments on 6th July, 3:00 pm. Also, the overall discussion volume decreases slightly over time. In terms of the location distribution, North America has a higher involvement in the discussions, but a variety of other countries located in different continents also talk about Shopify. For the sources, the most popular source is Twitter web client, followed by Twitter for iPhone and Twitter for Android, and WordPress.com.

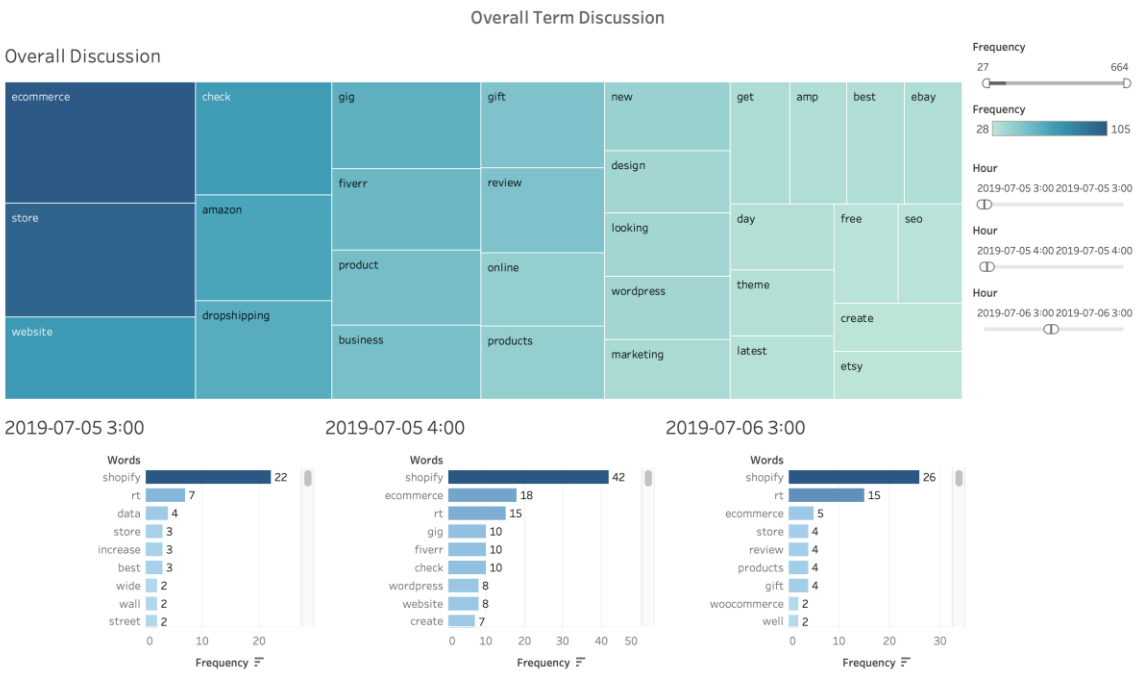


Illustration 22 Overall Term Discussion Dashboard

In addition to the overview of sentiments, there is also a need to know what people are talking about Shopify. To meet this need, a second dashboard is made called Overall Term Discussion which is shown in Illustration 22. The tree map on the top shows the term frequency of overall discussion based on all the available tweets of Shopify. The darker the color is, the more frequent that the term being mentioned in the tweets. We can see that the top ones are e-commerce, store, website, check, and Amazon, dropshipping, etc. Meanwhile, it's also good to know the term popularity on an hourly basis. The three bar charts at the bottom of the dashboard display the information for us by listing the terms from most frequent to less frequent for three individual hours which facilitates the comparison of the terms between hours. For example, by comparing the one for 5th July, 3:00 with the one for 5th July, 4:00, we can see how the term frequency changes over time.

Also, by comparing the one for 5th July, 3:00 and the one for 6th July, 3:00, it is possible to see the difference of the same hour from a different day.

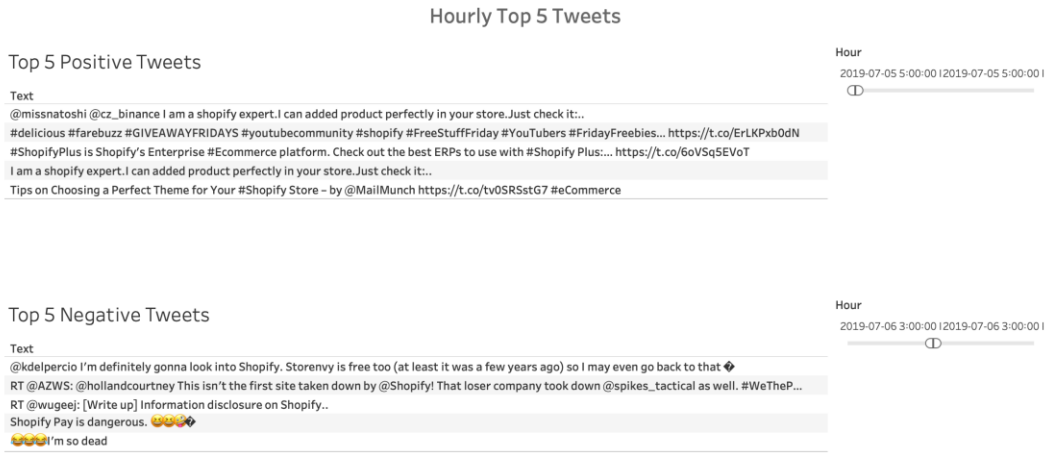


Illustration 23 Hourly Top Tweets Dashboard

The third dashboard is about hourly top tweets display which is shown in Illustration 23. Previously, we can see that there are spikes on the overall trend for positive and negative sentiments. As a marketing or public relationship people, there is a need to know what causes these spikes. We choose the hour for the two spikes and showcase the top 5 most positive and negative tweets here to give the users a general idea of what people are talking about during these hours. From the tweets, we can see that the positive spike (5th July, 5:00) is from the tweets that are talking about the building a better Shopify store. On the other hand, the negative spike (6th July, 3:00) is about a sudden site taken down of Shopify which causes complaints from the merchants.

Overall, we use a use case to illustrate how to utilize our proposed pipeline to apply Twitter sentiment analysis with "Shopify" as the keyword. It is clear that how the final results can benefit marketers in different companies and institutions by providing different keywords. For marketing purpose or maintaining public relation, marketers have a need to monitor what customers said on social media about their brand or companies. Some basic

metrics including the number of views, clicks, comments, likes, shares could provide a general view about the popularity of a specific topic related the brand or company. While after applying sentiment analysis, marketers can go beyond the metrics focus on quantity, but get a deeper understanding about the loves and hates of their customers, which is a better reference before taking any further actions. Specifically, if the marketers have the opportunities to know whether their customers have positive or negative sentiment towards the product they launched, the topic they created, or the activity they held, they can have a better understanding of the preference of the customers and make adjustments timely to improve their products or services.

6.2 Pipeline analysis and discussion

So far, we have introduced the details of our proposed pipeline, and utilize a use case to show how to apply our pipeline to solve a business problem. In this section, we want to review the whole pipeline and discuss some main benefit of using our proposed pipeline.

In the motivation, we mention that our goal is to build a sentiment analysis pipeline that is mostly automated, configurable, and can be updated periodically, which are all achieved in the pipeline design.

Firstly, we try to make every step automated when we design data collection, sentiment classification, and the data processing steps in the use case. In previous works, few studies consider automating the whole pipeline from data collection to sentiment classification. Normally, the model they proposed is based on some human-labeled datasets which will require extra human effort if it needs to be updated, or some datasets built by noisy labels but with a model not designed to run periodically. As a result, the previous models seldom consider building a highly automated pipeline but require human

effort for maintaining from time to time. In contrast, it is clear that the method we proposed to build training data based on Twitter Streaming API and noisy labels require nearly no human involvement. Also, the steps in sentiment classification model building are also mostly automated. For topic modeling, we utilize three different topic models and coherence scores to determine the best model in an automated way; We employ K-means clustering to assign the final cluster label to the tweets, and the only part require human judgment is to decide the number of clusters based on within-cluster sum of square, which we think is a good way to control the final number of clusters just in case a relatively big topic number is selected in the topic modeling step which would separate the training dataset into too many subgroups; For the use case, we discuss the details about how to ensure the data freshness in terms of the training dataset, classification model, and front-end dashboard and make model performance evaluation automatically. Overall, our proposed pipeline is mostly automated for all the steps which align with our final goal.

Secondly, our propose pipeline is configurable, which allows it to be applied to different questions. For data collection in our study, the auto-labeling is based on 42 commonly used graphic emojis which we consider is a relatively complete list for the emojis with distinguishable sentiment orientations. While the other users can easily change this emojis collection based on their own problem, by adding more emojis or narrow down the scope. This change can be easily applied by adding or removing Unicode of the emojis that used as the filter of Twitter Streaming API as shown in Illustration 24. For sentiment classification, three topic models are used in our pipeline and for the two which require topic number as preliminary, we parameterize the minimum and maximum topic number required so that the user can adjust the topic

number scope based on their own needs; For all the topic models, parameters (no_below and no_above) are provided to define the words will be included in topic modeling; As for clustering, the minimum and maximum cluster number can be decided by the use and a graph will be shown about the within-cluster sum of square of K-means for different numbers of clusters, which gives users the flexibility to determine the final number of clusters; The users can also switch between resampling, feature extraction, feature representative, feature selection modes, feature used, and classifier used. A code

```
twitter_stream.filter(
                      track=[u"\U0001F600",
                             u"\U0001F601",
                             u"\U0001F603",
                             u"\U0001F604",
                             u"\U0001F606",
                             u"\U0001F609",
                             u"\U0001F60A",
                             u"\U0001F60C",
                             u"\U0001F60B",
                             u"\U0001F60D",
                             u"\U0001F60E",
                             u"\U0001F60F",
                             u"\U0001F617",
                             u"\U0001F618",
                             u"\U0001F619",
                             u"\U0001F61A",
                             u"\U0001F62C",
                             u"\U0001F638",
                             u"\U0001F63A",
                             u"\U0001F63B",
                             u"\U0001F63C",
                             u"\U0001F63D"
                             ],
                      languages=['en']
)
```

Illustration 24 Example Code of Applying Emojis Filter for Data Collection

```
def main(self,
         no_below=5, no_above=0.4,
         lda_min_topic_num=3, lda_max_topic_num=30,
         lsi_min_topic_num=3, lsi_max_topic_num=30,
         min_cluster_number=2, max_cluster_number=15,
         resampling_mode='r_under_s',
         feature_extraction_mode = 'unigram', bigram_min_count=10,
         feature_represent_mode='tfidf',feature_selection_mode='chi2', top_n_feature=20000,
         classifier = 'logistic_regression',
         show_sample_tweets_head=15,
         feature_mode ='ngram_and_lexicon'):
```

Illustration 25 Example Code of Main Function for Sentiment Classification

snippet is shown in Illustration 25. Apart from the available parameters, addition or deletion or revision to the current pipeline is also doable. For example, LDA, LSI, and HDP are used as the topic models in our pipeline. While if another topic model is expected

to be added into the pipeline, it can be added without impacting any upstream and downstream. This is also applicable for changes on clustering and classification algorithms. For the front-end dashboard built based on Tableau, dashboard management and revision can be achieved without coding; For end users, filters are provided on the dashboard to focus on the parts they are most. Overall, the way we design each step is modular and loosely coupled, which allows the whole pipeline configurable.

Thirdly, we use a use case to explain how our proposed pipeline can be updated periodically, which is essential to drive a business application. The details about how to process streaming data from Twitter's API to build training dataset and update the classification models, and how to keep receiving the brand-related tweets in a continuous style and refresh the dashboard in an hourly basis are all introduced in Chapter 6.1. We hope all the details can give readers a better understanding of why and how to make our proposed pipeline updated periodically. Overall, our proposed pipeline meets the goals we planned.

Overall, it is clear that there are several benefits of using our pipeline instead of the existing tools. First of all, you can have a deep understanding of the pipeline design. For most of the existing tools, there may be documents about what methods (like, lexicon based or machine learning based) is applied in this model and how it works at a high level. While as a user, there is no space for you to understand the sentiment classification model driving the tools fully, nor can you explain why you get some specific results all the time. In contrast, instead of encapsulating the design details and providing the whole pipeline as a black box, our proposed pipeline is relatively understandable for every component in the pipeline. If the users are curious about why they got some specific results from the pipeline, it is possible for them to track what happened step by step

within the pipeline which would facilitate a better understanding and usage of our pipeline. Furthermore, you have full control on collecting training datasets, building sentiment classification models, and applying the model in a daily basis for your problem since the whole pipeline is configurable. Some current tools don't provide any ways to make improvements when you observe some misclassification issue which means these kinds of models cannot evolve over time. While our proposed pipeline is highly configurable, which allows it to be tuned easily, and also mostly automated, which allows rapid iteration. Last but not least, the design of our pipeline considers tracking the sentiment trend, which is a missing part in almost all the free online tools. By keep tracking the tweets based on the provided keyword, our pipeline can provide a better view of the sentiment changes over time, which facilitates hourly or daily sentiment comparison and anomaly detection.

So far, our proposed pipeline meets all the needs and goals we planned. Previously, we have seen some models proposed by academics which have potential to achieve good performance but are hard to be applied in real problems due to its complexity to understand and tune; Also, we have seen some free commercial tools perform poorly and can hardly be used directly in business applications. We hope our proposed pipeline could benefit small and medium companies and institutions and could be used as a free pipeline to solve their real business problems.

Chapter 7: Conclusion and Future Work

7.1 Conclusion

In this study, we focus on Twitter sentiment analysis which is a popular sub-topic under natural language processing, especially for the past several years. The goal of our thesis is to build a sentiment analysis pipeline which can be utilized to solve real business questions for marketing or public relation analysis purpose by companies or institutions. Although an extensive amount of works has been done for Twitter sentiment analysis aiming at subtasks from different steps or different levels, there are still some gaps to fill.

From the perspective of data collection, most updated training datasets are required to train a model due to the changing language style of tweets; While the changing policy of Twitter makes it much harder to redistribute labeled datasets which leads to a lack of publicly available datasets; In addition, the impact of time effect in Twitter specific datasets is seldomly considered in previous works; Also, the current way to build a training datasets only consider using string emoticons which is not traceable and manageable. From the perspective of sentiment classification model, there is a lack of awareness that it is essential to uncover the semantic topics when using training dataset annotated by noisy labels since there is no topic limitation in the training dataset; Furthermore, there is no existing way to choose the best topic number in order to select the best topic models from a group of topic models have been built. For the perspective of use case studies, few previous works consider the scenarios when applying the proposed models to solve business problems in a daily basis which require the model to be run periodically. Based on all the gaps mentioned above, we are motivated to build a

pipeline that includes the major components in Twitter sentiment analysis and make corresponding improvements based on previous works.

To fill these gaps, we make some improvements based on previous works in this study. For data collection, we proposed a new data collection method by utilizing the Twitter Streaming API and graphic emojis. This method will not be limited to Twitter's changing policy, also has the ability to collect data in a real-time which makes the training datasets easy to evolve and can deal with the time effect in the Twitter training dataset. Meanwhile, we also build a list of 42 commonly used graphic emojis in terms of positive and negative sentiments which can facilitate the building of training datasets in an automated way. For Twitter sentiment classification, we proposed a novel topic-model based hybrid sentiment classification model as the following step to consume the training datasets we built. Topic modeling is employed as the first step after data preparation to uncover the latent semantic topics in our training datasets so that the lack of topic limitation in tweets won't be a big issue when building supervised learning classifiers later on. We also propose a new method to select the best model from a group of built topic models with the help of the coherence score. By combining unsupervised approaches (topic modeling, k-means clustering), and supervised learning (Random Forest, Logistic Regression algorithm) together, our proposed model achieves a 78.54 F-measure and outperforms the previous work [28] that using the same test dataset. For use case study, we provide a use case study from the perspective of marketers working for Shopify, to illustrate the data processing details about how to apply our proposed pipeline in a daily basis to solve a real problem. Overall, we proposed a comprehensive pipeline

for Twitter sentiment analysis which is mostly automated and configurable, which has the potential to be used for marketing or public relation analysis purpose on a daily basis.

7.2 Future Works

Apart from proposing a Twitter sentiment analysis pipeline, this study also raises several questions that can be investigated further in further works.

Currently, we consider sentiment analysis as a binary classification based on the pipeline, which does not consider neutral as a class in the result. While in reality, the existence of neutral tweets is not a surprise to us. We did not include neutral tweet is due to the consideration of making the whole pipeline mostly automated but acquiring the newest neutral tweets in an automated way without human efforts is still a challenging task for us at this moment. Wikipedia has been considered as a source of building neutral sentiment analysis dataset, while since the language styles between Wikipedia and Twitter have a fundamental difference, the utility of using Wikipedia need to be tested; News official accounts in Twitter is considered as another potential source for neutral tweets. The benefit of this approach is all the neutral part is built on Twitter which aligns with the positive and negative datasets; While one concern is whether the volume of neutral tweets collected this way will match the volumes for positive and negative tweets; And also, how to verify the selection of news accounts is good enough for building the datasets. We need to spend some time in future work to determine the best way to build neutral dataset in an automated way.

In addition, we utilize a use case to illustrate how to apply our proposed pipeline on a real business problem, while the final goal is to make the whole pipeline a product which requires the user less work to run, schedule, and maintain. Currently, we only discuss the steps in the use case from the perspective of data. In further work, we will also discuss

the implementation details for our proposed pipeline.

Bibliography or References

[1] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining text data*, Springer, 2012, pp. 415–463.

[2] M. Tsytsarau and T. Palpanas, "Survey on mining subjective data on the web," *Data Min. Knowl. Discov.*, vol. 24, no. 3, pp. 478–514, 2012.

[3] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguist.*, vol. 37, no. 2, pp. 267–307, 2011.

[4] A. Giachanou and F. Crestani, "Like it or not: A survey of twitter sentiment analysis methods," *ACM Comput. Surv.*, vol. 49, no. 2, p. 28, 2016.

[5] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent semantic indexing: A probabilistic analysis," *J. Comput. Syst. Sci.*, vol. 61, no. 2, pp. 217–235, 2000.

[6] N. Aston, J. Liddle, and W. Hu, "Twitter sentiment in data streams with perceptron," *J. Comput. Commun.*, vol. 2, no. 03, p. 11, 2014.

[7] N. F. F. Da Silva, E. R. Hruschka, and E. R. Hruschka Jr, "Tweet sentiment analysis with classifier ensembles," *Decis. Support Syst.*, vol. 66, pp. 170–179, 2014.

[8] N. A. Diakopoulos and D. A. Shamma, "Characterizing debate performance via aggregated twitter sentiment," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 1195–1198.

[9] X. Hu, L. Tang, J. Tang, and H. Liu, "Exploiting social relations for sentiment analysis in microblogging," in *Proceedings of the sixth ACM international*

conference on Web search and data mining, 2013, pp. 537–546.

[10] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge, "Twitter polarity classification with label propagation over lexical links and the follower graph," in *Proceedings of the First workshop on Unsupervised Learning in NLP*, 2011, pp. 53–63.

[11] N. Saleena, "An Ensemble Classification System for Twitter Sentiment Analysis," *Procedia Comput. Sci.*, vol. 132, pp. 937–946, 2018.

[12] T.-J. Lu, "Semi-supervised microblog sentiment analysis using social relation and text similarity," in *2015 International conference on big data and smart computing (BigComp)*, 2015, pp. 194–201.

[13] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N Proj. Report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[14] S. Petrović, M. Osborne, and V. Lavrenko, "The edinburgh twitter corpus," in *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, 2010, pp. 25–26.

[15] E. Kouloumpis, T. Wilson, and J. Moore, "Twitter sentiment analysis: The good the bad and the omg!," in *Fifth International AAAI conference on weblogs and social media*, 2011.

[16] N. Naveed, T. Gottron, J. Kunegis, and A. C. Alhadi, "Searching microblogs: coping with sparsity and document quality," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 183–188.

[17] B. O'Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith, "From

tweets to polls: Linking text sentiment to public opinion time series," in *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[18] D. Davidov, O. Tsur, and A. Rappoport, "Enhanced sentiment learning using twitter hashtags and smileys," in *Proceedings of the 23rd international conference on computational linguistics: posters*, 2010, pp. 241–249.

[19] A. Asiaee T, M. Tepper, A. Banerjee, and G. Sapiro, "If you are happy and you know it... tweet," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 1602–1606.

[20] H. Saif, M. Fernandez, Y. He, and H. Alani, "Evaluation datasets for Twitter sentiment analysis: a survey and a new dataset, the STS-Gold," 2013.

[21] H. Saif, Y. He, M. Fernandez, and H. Alani, "Contextual semantics for sentiment analysis of Twitter," *Inf. Process. Manag.*, vol. 52, no. 1, pp. 5–19, 2016.

[22] O. Araque, I. Corcuera-Platas, J. F. Sanchez-Rada, and C. A. Iglesias, "Enhancing deep learning sentiment analysis with ensemble techniques in social applications," *Expert Syst. Appl.*, vol. 77, pp. 236–246, 2017.

[23] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets," *arXiv Prepr. arXiv1308.6242*, 2013.

[24] S. Kiritchenko, X. Zhu, and S. M. Mohammad, "Sentiment analysis of short informal texts," *J. Artif. Intell. Res.*, vol. 50, pp. 723–762, 2014.

[25] T. Chalothom and J. Ellman, "Simple approaches of sentiment analysis via ensemble learning," in *information science and applications*, Springer, 2015, pp. 631–639.

[26] B. Xiang and L. Zhou, "Improving twitter sentiment analysis with topic-based

mixture modeling and semi-supervised training," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014, vol. 2, pp. 434–439.

[27] P. Balage Filho and T. Pardo, "NILC_USP: A hybrid system for sentiment analysis in twitter messages," in *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2013, vol. 2, pp. 568–572.

[28] N. Malandrakis, A. Kazemzadeh, A. Potamianos, and S. Narayanan, "SAIL: A hybrid approach to sentiment analysis," in *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2013, vol. 2, pp. 438–442.

[29] S. M. Jiménez-Zafra, M. T. Martín-Valdivia, E. Martínez-Cámara, and L. A. Ureña-López, "Combining resources to improve unsupervised sentiment analysis at aspect-level," *J. Inf. Sci.*, vol. 42, no. 2, pp. 213–229, 2016.

[30] M. Hagen, M. Potthast, M. Büchner, and B. Stein, "Twitter sentiment detection via ensemble classification using averaged confidence scores," in *European Conference on Information Retrieval*, 2015, pp. 741–754.

[31] D. Tang, F. Wei, B. Qin, T. Liu, and M. Zhou, "Coooolll: A deep learning system for twitter sentiment classification," in *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, 2014, pp. 208–212.

[32] A. Severyn and A. Moschitti, "Twitter Sentiment Analysis with Deep

Convolutional Neural Networks," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 959–962.

[33] J. D. Prusa, T. M. Khoshgoftaar, and A. Napolitano, "Using feature selection in combination with ensemble learning techniques to improve tweet sentiment classification performance," in *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2015, pp. 186–193.

[34] M. Cliche, "BB_twtr at SemEval-2017 task 4: twitter sentiment analysis with CNNs and LSTMs," *arXiv Prepr. arXiv1704.06125*, 2017.

[35] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 168–177.

[36] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, "Sentiment strength detection in short informal text," *J. Am. Soc. Inf. Sci. Technol.*, vol. 61, no. 12, pp. 2544–2558, 2010.

[37] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining.," in *Lrec*, 2010, vol. 10, no. 2010, pp. 2200–2204.

[38] A. Esuli and F. Sebastiani, "SentiWordNet: a high-coverage lexical resource for opinion mining," *Evaluation*, vol. 17, no. 1, p. 26, 2007.

[39] S. M. Mohammad and P. D. Turney, "Nrc emotion lexicon," *Natl. Res. Counc. Canada*, 2013.

[40] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and

emotions in language," *Lang. Resour. Eval.*, vol. 39, no. 2–3, pp. 165–210, 2005.

[41] M. Z. Asghar, F. M. Kundi, S. Ahmad, A. Khan, and F. Khan, "T-SAF: Twitter sentiment analysis framework using a hybrid classification scheme," *Expert Syst.*, vol. 35, no. 1, p. e12233, 2018.

[42] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment strength detection for the social web," *J. Am. Soc. Inf. Sci. Technol.*, vol. 63, no. 1, pp. 163–173, 2012.

[43] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in *LREc*, 2010, vol. 10, no. 2010, pp. 1320–1326.

[44] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of twitter data," in *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, 2011, pp. 30–38.

[45] Y. Zhang, D. Song, P. Zhang, X. Li, and P. Wang, "A quantum-inspired sentiment representation model for twitter sentiment analysis," *Appl. Intell.*, pp. 1–16, 2019.

[46] J. Prusa, T. M. Khoshgoftaar, and D. J. Dittman, "Using ensemble learners to improve classifier performance on tweet sentiment data," in *2015 IEEE International Conference on Information Reuse and Integration*, 2015, pp. 252–257.

[47] Y. Wan and Q. Gao, "An ensemble sentiment classification system of twitter data for airline services analysis," in *2015 IEEE international conference on data mining workshop (ICDMW)*, 2015, pp. 1318–1325.

[48] J. Prusa, T. M. Khoshgoftaar, and A. Napolitano, "Utilizing ensemble, data sampling and feature selection techniques for improving classification performance on tweet sentiment data," in *2015 IEEE 14th International*

Conference on Machine Learning and Applications (ICMLA), 2015, pp. 535–542.

[49] J. Prusa, T. M. Khoshgoftaar, D. J. Dittman, and A. Napolitano, "Using random undersampling to alleviate class imbalance on tweet sentiment data," in *2015 IEEE international conference on information reuse and integration*, 2015, pp. 197–202.

[50] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep convolution neural networks for Twitter sentiment analysis," *IEEE Access*, vol. 6, pp. 23253–23260, 2018.

[51] Y. Arslan, D. Küçük, and A. Birturk, "Twitter Sentiment Analysis Experiments Using Word Embeddings on Datasets of Various Scales," in *International Conference on Applications of Natural Language to Information Systems*, 2018, pp. 40–47.

[52] L. Naji, "Twitter Sentiment Analysis Training Corpus (Dataset)," 2012.

[53] F. Godin, B. Vandersmissen, W. De Neve, and R. Van de Walle, "Multimedia Lab $@ $ ACL WNUT NER Shared Task: Named Entity Recognition for Twitter Microposts using Distributed Word Representations," in *Proceedings of the Workshop on Noisy User-generated Text*, 2015, pp. 146–153.

[54] F. Abid, M. Alam, M. Yasir, and C. Li, "Sentiment analysis through recurrent variants latterly on convolutional neural network of Twitter," *Futur. Gener. Comput. Syst.*, vol. 95, pp. 292–308, 2019.

[55] W. Ding, X. Song, L. Guo, Z. Xiong, and X. Hu, "A novel hybrid HDP-LDA model for sentiment analysis," in *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 01*, 2013, pp. 329–336.

[56] S. Poria, I. Chaturvedi, E. Cambria, and F. Bisio, "Sentic LDA: Improving on

LDA with semantic similarity for aspect-based sentiment analysis," in *2016 international joint conference on neural networks (IJCNN)*, 2016, pp. 4465–4473.

[57] Z. Liu, X. Dong, Y. Guan, and J. Yang, "Reserved self-training: A semi-supervised sentiment classification method for chinese microblogs," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 2013, pp. 455–462.

[58] W. Baugh, "bwbaugh: Hierarchical sentiment analysis with partial self-training," in *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2013, vol. 2, pp. 539–542.

[59] J. Zhao, M. Lan, and T. Zhu, "ECNU: Expression-and message-level sentiment orientation classification in Twitter using multiple effective features," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 259–264.

[60] S. Liu *et al.*, "Co-training and visualizing sentiment evolvement for tweet events," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 105–106.

[61] N. Ponomareva and M. Thelwall, "Semi-supervised vs. cross-domain graphs for sentiment analysis," in *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, 2013, pp. 571–578.

[62] K. Kim and J. Lee, "Sentiment visualization and classification via semi-supervised nonlinear dimensionality reduction," *Pattern Recognit.*, vol. 47, no. 2, pp. 758–

768, 2014.

[63] H. Suresh, “An unsupervised fuzzy clustering method for twitter sentiment analysis,” in *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, 2016, pp. 80–85.

[64] S. Riaz, M. Fatima, M. Kamran, and M. W. Nisar, “Opinion mining on large scale data using sentiment analysis and k-means clustering,” *Cluster Comput.*, pp. 1–16, 2017.

[65] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu, “Combining lexicon-based and learning-based methods for Twitter sentiment analysis,” *HP Lab. Tech. Rep. HPL-2011*, vol. 89, 2011.

[66] N. Mittal, B. Agarwal, S. Agarwal, S. Agarwal, and P. Gupta, “A hybrid approach for twitter sentiment analysis,” in *10th international conference on natural language processing (ICON-2013)*, 2013, pp. 116–120.

[67] F. H. Khan, S. Bashir, and U. Qamar, “TOM: Twitter opinion mining framework using hybrid classification scheme,” *Decis. Support Syst.*, vol. 57, pp. 245–257, 2014.

[68] M. Ghiassi, J. Skinner, and D. Zimbra, “Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network,” *Expert Syst. Appl.*, vol. 40, no. 16, pp. 6266–6282, 2013.

[69] S. Poria, A. Gelbukh, B. Agarwal, E. Cambria, and N. Howard, “Sentic Demo: A hybrid concept-level aspect-based sentiment analysis toolkit,” *ESWC 2014*, 2014.

[70] A. Hasan, S. Moin, A. Karim, and S. Shamshirband, “Machine learning-based sentiment analysis for twitter accounts,” *Math. Comput. Appl.*, vol. 23, no. 1, p. 11,

2018.

[71] N. Al-Twairesh, H. Al-Khalifa, A. Alsalman, and Y. Al-Ohali, “Sentiment analysis of arabic tweets: Feature engineering and a hybrid approach,” *arXiv Prepr. arXiv1805.08533*, 2018.

[72] V. Ramanathan and T. Meyyappan, “Twitter Text Mining for Sentiment Analysis on People’s Feedback about Oman Tourism,” in *2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, 2019, pp. 1–5.

[73] S. M. Nagarajan and U. D. Gandhi, “Classifying streaming of Twitter data based on sentiment analysis using hybridization,” *Neural Comput. Appl.*, pp. 1–9, 2018.

[74] L. Wang, J. Niu, and S. Yu, “SentiDiff: Combining Textual Information and Sentiment Diffusion Patterns for Twitter Sentiment Analysis,” *IEEE Trans. Knowl. Data Eng.*, 2019.

[75] B. Pang and L. Lee, “A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts,” in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, 2004, p. 271.

[76] G. Ganu, N. Elhadad, and A. Marian, “Beyond the stars: improving rating predictions using review text content.,” in *WebDB*, 2009, vol. 9, pp. 1–6.

[77] N. Malandrakis, A. Potamianos, G. Evangelopoulos, and A. Zlatintsi, “A supervised approach to movie emotion tracking,” in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2011, pp. 2376–2379.

[78] P. D. Turney and M. L. Littman, “Unsupervised learning of semantic orientation

from a hundred-billion-word corpus," *arXiv Prepr. cs/0212012*, 2002.

[79] Y.-S. Chen, L.-H. Chen, and Y. Takama, "Proposal of lda-based sentiment visualization of hotel reviews," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, 2015, pp. 687–693.

[80] M. Araújo, P. Gonçalves, M. Cha, and F. Benevenuto, "iFeel: a system that compares and combines sentiment analysis methods," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 75–78.

[81] M. L. D. Araujo *et al.*, "ifeel 2.0: A multilingual benchmarking system for sentence-level sentiment analysis," in *Tenth International AAAI Conference on Web and Social Media*, 2016.

[82] Y. Zhao, B. Qin, T. Liu, and D. Tang, "Social sentiment sensor: a visualization system for topic detection and topic sentiment analysis on microblog," *Multimed. Tools Appl.*, vol. 75, no. 15, pp. 8843–8860, 2016.

[83] Z. Zhao, Y. Zhang, C. Li, L. Ning, and J. Fan, "A system to manage and mine microblogging data," *J. Intell. Fuzzy Syst.*, vol. 33, no. 1, pp. 315–325, 2017.

[84] N. H. Mahadzir, M. F. Omar, and M. N. M. Nawi, "A Sentiment Analysis Visualization System for the Property Industry," *Int. J. Technol.*, vol. 9, no. 8, pp. 1609–1617, 2018.

[85] U. Yaqub, N. Sharma, R. Pabreja, S. Chun, V. Atluri, and J. Vaidya, "Analysis and visualization of subjectivity and polarity of twitter location data," in *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, 2018, p. 67.

[86] Y. H. Gu *et al.*, "Sentiment analysis and visualization of Chinese tourism blogs

and reviews," in *2018 International Conference on Electronics, Information, and Communication (ICEIC)*, 2018, pp. 1–4.

[87] S. Urologin, "Sentiment Analysis, Visualization and Classification of Summarized News Articles: A Novel Approach," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 8, pp. 616–625, 2018.

[88] M. Al-Ghalibi, A. Al-Azzawi, and K. Lawonn, "NLP based sentiment analysis for Twitter's opinion mining and visualization," in *Eleventh International Conference on Machine Vision (ICMV 2018)*, 2019, vol. 11041, p. 110412A.

[89] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 English lemmas," *Behav. Res. Methods*, vol. 45, no. 4, pp. 1191–1207, 2013.

[90] P. S. Dodds, K. D. Harris, I. M. Kloumann, C. A. Bliss, and C. M. Danforth, "Temporal patterns of happiness and information in a global social network: Hedonometrics and Twitter," *PLoS One*, vol. 6, no. 12, p. e26752, 2011.

[91] S. Liu, Y. Wang, J. Zhang, C. Chen, and Y. Xiang, "Addressing the class imbalance problem in twitter spam detection using ensemble learning," *Comput. Secur.*, vol. 69, pp. 35–49, 2017.

[92] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Icml*, 1997, vol. 97, no. 412–420, p. 35.

[93] J. D. Altman *et al.*, "Phenotypic analysis of antigen-specific T lymphocytes," *Science (80-.).*, vol. 274, no. 5284, pp. 94–96, 1996.

[94] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Am. Soc. Inf. Sci.*, vol. 41, no. 6, pp.

391–407, 1990.

[95] T. Hofmann, “Probabilistic latent semantic indexing,” in *ACM SIGIR Forum*, 2017, vol. 51, no. 2, pp. 211–218.

[96] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *J. Mach. Learn. Res.*, vol. 3, no. Jan, pp. 993–1022, 2003.

[97] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, “Hierarchical Dirichlet Processes,” *J. Am. Stat. Assoc.*, vol. 101, no. 476, pp. 1566–1581, Dec. 2006.

[98] J. Chuang, C. D. Manning, and J. Heer, “Termite: Visualization techniques for assessing textual topic models,” in *Proceedings of the international working conference on advanced visual interfaces*, 2012, pp. 74–77.

[99] J. Chuang, S. Gupta, C. Manning, and J. Heer, “Topic model diagnostics: Assessing domain relevance via topical alignment,” in *International Conference on Machine Learning*, 2013, pp. 612–620.

[100] C. Sievert and K. Shirley, “LDAvis: A method for visualizing and interpreting topics,” in *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, 2014, pp. 63–70.

[101] J. Chang, S. Gerrish, C. Wang, J. L. Boyd-Graber, and D. M. Blei, “Reading tea leaves: How humans interpret topic models,” in *Advances in neural information processing systems*, 2009, pp. 288–296.

[102] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin, “Automatic evaluation of topic coherence,” in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*,

2010, pp. 100–108.

[103] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing semantic coherence in topic models," in *Proceedings of the conference on empirical methods in natural language processing*, 2011, pp. 262–272.

[104] N. Aletras and M. Stevenson, "Evaluating topic coherence using distributional semantics," in *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, 2013, pp. 13–22.

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

[106] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[107] J. Berkson, "A statistically precise and relatively simple method of estimating the bio-assay with quantal response, based on the logistic function," *J. Am. Stat. Assoc.*, vol. 48, no. 263, pp. 565–599, 1953.

[108] N. Chinchor, "MUC-4 evaluation metrics," in *Proceedings of the 4th conference on Message understanding*, 1992, pp. 22–29.

[109] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

[110] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *advances in neural information processing systems*, 2010, pp. 856–

864.

[111] C. Wang, J. Paisley, and D. Blei, “Online variational inference for the hierarchical Dirichlet process,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 752–760.

[112] D. Arthur and S. Vassilvitskii, “k-means++: The advantages of careful seeding,” in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 1027–1035.