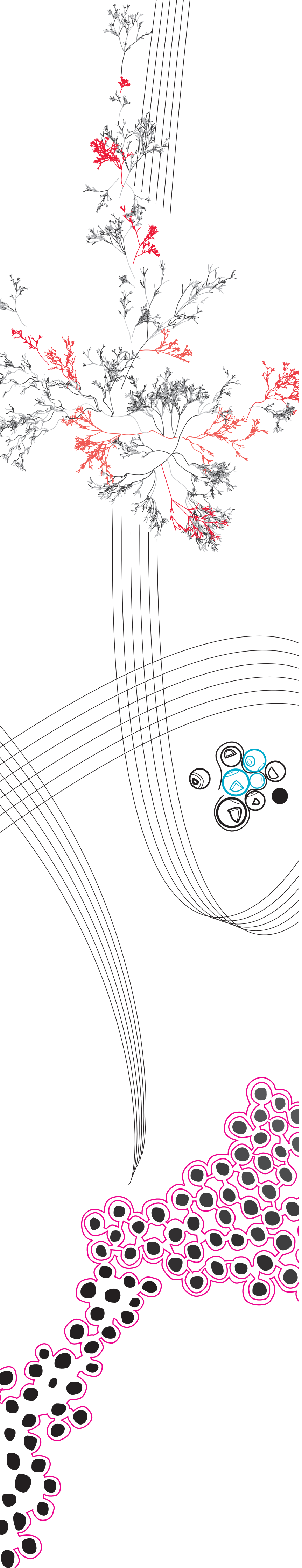

MASTER THESIS

Composing a more complete and relevant Twitter dataset

Han van der Veen

Faculty of Electrical Engineering, Mathematics and Computer Science
Databases Research Group

Committee:
Dr. Djoerd Hiemstra (DB/CTIT)
Dr. Robin Aly (DB/CTIT)
Tijs van den Broek (NIKOS/BMS)

August 11, 2015

UNIVERSITY OF TWENTE.

Abstract

Social data is widely used by many researchers. Facebook, Twitter and other social networks are producing huge amounts of social data. This social data can be used for analyzing human behavior. Social datasets are typically created by a hashtag, however not all relevant data includes the hashtag. A better overview can be constructed with more data. This research is focusing on creating a more complete and relevant dataset. Using additional keywords for finding more relevant tweets and a filtering mechanism to filter out the irrelevant tweets. Three additional keywords methods are proposed and evaluated. One based on word frequency, one on probability of word in a dataset and the last method is using estimates about the volume of tweets. Two classifiers are used for filtering Tweets. A Naïve Bayes classifier and a Support Vector Machine classifier are compared. Our method increases the size of the dataset with 105%. The average precision was reduced from 95% of only using a hashtag to 76% for a resulting dataset. These evaluations were executed on two TV-Shows and two sport events. A tool was developed that automatically executes all parts of the program. As input a specific hashtag of an event is required and using the hash will output a more complete and relevant dataset than using the original hashtag. This is useful for social researchers that uses Tweets, but also other researchers that uses Tweets as their data.

Preface

This master thesis is a result of my final project conducted for the Database group of the University of Twente. I always had an interest in the web and with all the data it contains. It all started with my first database in MySQL and now I was able to work with loads of Tweets. I have enjoyed working at this project en the final program can be used by everyone at `https://github.com/haneev/TweetRetriever`.

Many thanks to Djoerd Hiemstra who has guided me all the way. And to Tijs van den Broek always giving feedback and helped me to make this research relevant to the #datagrant project. And thanks to Robin Aly which provided me useful feedback on my whole research.

I hope you will enjoy reading this research.

Han van der Veen
August 11, 2015

Contents

List of symbols and abbreviations

JSON JavaScript Object Notation.

Tweet Short message from Twitter

HTML HyperText Markup Language

Firehose Live-stream of all Tweets on Twitter

Gardenhose Stream of tweets. About 1% of the total Tweets.

API Application Programmable Interface

Streaming API API of Twitter that allows subscribing to certain keywords. Only tweets using the keywords are returned.

Bayes Used to reference to Multinomial Naïve Bayes

SVM Support Vector Machine.

C4.5 Commonly used decision tree algorithm

Hashtag A hashtag is a word prefixed with a # used for relating Tweets to certain topics.

Token A token is a part of a sentence created by a tokenizer, usually a word.

1

Introduction

Huge amounts of data are produced by Facebook, WhatsApp, Twitter and social companies on the web. More than 500 million Tweets are posted every day [4]. Every message contains information, for example an opinion, shared link, reaction, call for action or something else. In this big pile of data information can be extracted. For instance, a topic the user is referring to in a Tweet. Finding actual information in each message is hard. However, by analyzing the data, useful information can be found using machine learning or other methods.

Twitter is a social network that allows people to post messages on a wall. Twitter users can follow other Twitter users. Each Tweet can be marked as favorite, retweeted or replied on. A retweet is sharing a copy of the original post on your own wall, because you want to show that message to your own followers.

Data is very important for many researchers. When you have data from measurements you ensure that the dataset is reliable. For researchers in information retrieval a dataset of document used on the internet is required for their classification and searching tasks. These datasets should be reliable and a representation of the real world. Tweets provides information about social interactions, which is valuable for social research [37]. This social data mainly covers user actions. For example, posting messages, liking pages, sharing messages. This data is publicly available and can be used to see if certain analogies can be made from the data. You can think of predicting flu trends [13] or predicting the outcome of elections [43] although there is controversy in predicting elections [36, 23].

In this thesis the data will be collected from Twitter. Twitter has granted the University of Twente access to all historical data related to *cancer awareness campaigns* [3] (called the Twitter #datagrant). The #datagrant team is looking at campaigns against cancer. Such as, Movember against prostate cancer, PinkRibbon against breast cancer and several other campaigns. The effectiveness of the campaigns are determined by looking at the behavior of the users before, during and after the running time of the campaign. To make reliable conclusions, an accurate dataset is required.

In this research the methods will be validated against data of the Twitter Streaming API. The Twitter streaming API allows to subscribe to certain keywords. All Tweets using those keywords will be returned. The data in this thesis will be about TV-shows and sport events. Nonetheless, the method should also work on *cancer awareness campaigns* and other domains. This thesis will research and develop a method for composing datasets for researchers who want accurate data from Twitter.

An accurate dataset consists of a complete and relevant set of tweets. The measure *precision* gives an indication of the relevancy and *recall* an indication of the completeness of the dataset. More about precision and recall in chapter 2.

1.1 Motivation

In research creating a reliable dataset can be challenging. Standardized datasets are used in order to solve the problem of a transparent and evenly distributed dataset. When data is used for research and conclusions needs to be made from it, the data should be reliable. Reliable means that the data is predictable and is relevant. Automatically creating a dataset for different purposes is also difficult, because each domain has a specific requirements. The need for a reliable dataset is relevant for everyone working with data, especially for people who are analyzing huge amounts of data, because manual labor cannot be applied on large datasets, due to time limitations and human error. An automatic approach is required in order to handle huge amount of data.

Currently, a lot of messages on the Internet are very short. Many text classifiers are evaluated against a standard dataset. For example, Reuters dataset of news articles have more characters than tweets or Facebook posts. A specialized method can improve classifiers for Tweets and other short messages.

Knowing what is relevant of a certain topic is useful for other purposes than creating a dataset. Filtering on a topic or filter out everything that is not related to a topic might be another application of this research.

1.2 Problem statement

Everyday around 500 million tweets are posted on Twitter[4]. Sometimes a topic is linked to a certain hashtag. In our example #movember is the hashtag for the campaign Movember. In the movember campaign every men who is supporting the campaign does not shave in the month November to be a visible supporter of the cause. To create a complete dataset tweets who are referring to mustache, but not using the hashtag movember might still be relate to the campaign movember. On the other hand, there are also words which have different meanings such as *cancer*. Which can mean the illness cancer or a sign in the stars. So, using more keywords increases the amount of possible relevant tweets, but reduces the recall. More about this in section 2.

Knowing whether a keyword is representative for a topic or not, is an issue. A keyword can be used a lot in a set of Tweets, but does that mean that the keyword is representative. It might be the case that the word *spoon* is linked to a campaign, because in a campaign TV-spot a spoon is used. However, the word spoon is also used in other contexts. Concluding, a word which is used in a campaign is not on itself representative for the campaign. Filtering out the irrelevant Tweets is another challenge. Both problems are addressed in this research.

In this research some questions are used for guidance.

RQ How to compose a more complete and relevant dataset on a topic for short messages?

Q1 How to create a more complete dataset on a topic by identifying additional keywords?

Q2 How to create a dataset of relevant Tweets on a topic by filtering the dataset?

Q3 How to combine filtering and identifying additional keywords in such a way that it will result in a more complete and relevant dataset?

Question *RQ* is the main research question what needs to be solved. Question *Q1* and *Q2* are sub-problems which splits the main research problem into two separate parts, namely a question about relevance and a question about completeness. Question *Q3* is combining the methods of *Q1* and *Q2* in order to improve the overall result. And *Q3* will build a method that can be trained and produce a reliable dataset that can be used by the #datagrant team or other researchers.

1.3 Goal

The goal of this research is to create a complete and relevant dataset, bootstrapped by one hashtag. When all data of the Movember campaign is required, the hashtag #movember is used to identify additional keywords in order to find all related Tweets. While receiving Tweets, each Tweet will be classified as relevant or irrelevant. The outcome will be a relevant and more complete dataset about Movember. Also a balance between the completeness (recall) and reliability (precision) needs to be found.

1.4 Organization

This thesis is structured by firstly giving some background information in chapter 2. Followed by a literature study in chapter 3. In chapter 4 *Q1* is addressed. Questions *Q2* and *Q3* are addressed respectively in chapter 5 and chapter 6. Concluding with discussion and conclusion in chapter 7.

2
Background

To understand this thesis some basic understanding about machine learning, information retrieval and evaluation methods is required.

2.1 Precision and Recall

While evaluating the designed methods the results are expressing in *precision* and *recall.* This section is based on chapter 8 of Manning et al. [35].

Precision is the fraction of retrieved documents that are relevant. For example, a search for pictures of airplanes. The query will be *airplane picture.* Only images of airplanes are relevant. All documents that are about airplanes, but are not images are not relevant. Precision gives us the fraction of the retrieved documents that are airplane pictures. Precision can be expressed by dividing the number of relevant documents by the total retrieved items.

$$Precision = \frac{retrieved_{relevant}}{retrieved} = P(relevant|retrieved) \tag{2.1}$$

Recall is the fraction of relevant documents that are retrieved. In the example of airplanes, searching for pictures of airplanes. The query still is *airplane pictures.* Recall is expressing how much documents that are pictures of airplanes. A recall of 1.0 indicates that **all** pictures of airplanes are retrieved and no documents are missed or skipped. Recall can be expressed by dividing the retrieved relevant (*rel*) items by the total existing relevant items.

$$Recall = \frac{retrieved_{relevant}}{relevant} = P(retrieved|relevant) \tag{2.2}$$

Another method for calculating precision and recall is using a truth table 2.1. With this table a retrieved document can be a *true positive* or a *false positive.* A true positive document is a document that is relevant and is retrieved, a success for the information retrieval system. A false positive document is a retrieved, but not relevant document. For not retrieved documents a document can be *false negative* or *true negative.* Where false negative is a relevant, but not retrieved document. The system have missed this document. A true negative is a non-relevant and not retrieved document. In a *good* information retrieval system, the amount of false positive and false negatives document is small.

	Relevant	Non-relevant
Retrieved	true positives (tp)	false positives (fp)
Not retrieved	false negatives (fn)	true negatives (tn)

Table 2.1: Table for naming relevant vs retrieved

Precision and Recall can also be calculated using table 2.1 and equation 2.3.

$$Precision = \frac{tp}{tp + fp} \tag{2.3}$$

$$Recall = \frac{tp}{tp + fn} \tag{2.4}$$

Accuracy is an overall result on all the available data and is the fraction of correctly identified documents.

$$Accuracy = \frac{\text{correctly identified documents}}{\text{all documents}} = \frac{tp + tn}{tp + fp + fn + tn} \tag{2.5}$$

To provide one score for comparison the F-measure was invented. This is a combination of precision and recall. When precision is improved and the loss of recall is less, the F1-measure will produce a higher number, because the overall result is better.

$$F_1\text{-measure} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.6}$$

Law of information retrieval There exists a relationship between precision and recall. When retrieving more documents (improving recall) the precision will be lower. And the other way around, when the precision is higher the recall will be lower. This is sometimes called the *law of information retrieval.*

2.2 Tasks

In information retrieval there are several common tasks. The most obvious task is *searching.* Finding documents that give answer to the user query. This includes parsing a dataset of documents, indexing documents and finding relevant documents. Widely known search engines are Google, Yahoo and Lucene.

Classification of documents is also a researched task. Classification of text, parsing images and other ways of automatically classifying data. A document can be of all formats. For instance, tweets, images, news-articles, raw-data files, videos or anything else. Classification is a method that will automatically classify a document into a certain class. For example, a binary classification, is this tweet about movember. The answer can be either yes or no. There are also multi class classifiers. Answering the classification question: what is the origin of this tweet. Any country in the world can be the answer. However, for countries the output is still predictable and finite. Another classification question is: what is the topic of this tweet? The answer can be any topic, because Twitter is all over the world. Other tasks are for example, clustering, finding patterns, finding non-patterns and so on.

2.3 Machine Learning

An area that automatically learns from some training data and applies on a new situations or documents, is called Machine Learning. Machine Learning can be applied on all sort of tasks. From reading brain signals and convert them to movements in a game or classifying text documents into classes. Machine learning is the term used for these apply by learning techniques.

Machine Learning can be supervised or unsupervised [29]. Supervised machine learning uses training data that are (manually) annotated in order to train the method. After training new situations are given to the method and it will automatically gives an results that is based on the training data.

Unsupervised has training data that is not annotated. The method will find relations and classes on itself. These results are unlabeled, but can still be used to classify documents for instance. It can see that the documents are about the same topic, however it does not know what the topic is.

There are several basic machine learning tasks [7]. This research is mainly focusing on classification. However, clustering could also be used when having a dataset and the topic is not known.

- In *classification*, training data with predefined classes are used to classify new data into classes. As described above.
- In *regression* the output are not continues instead of discrete. For instance, based on the received training values a parameter of a system was 0.78.
- In *clustering* the input is grouped into several clusters. The groups are not known beforehand and are generated by the learner.
- In *density estimation* the density will be determined by the learner. This can be useful to predict boundaries of a problem.
- *Dimensionality reduction* simplifies the problem space into a lower-dimensional space. This is useful for simplifying problems, in order to improve solving times of problems.

3

Related work

3.1 Introduction

This research examines the combination of choosing keywords and validating the choice of each keyword. In current research there are many approaches addressing finding additional keywords and a lot of research in classifiers. In this section we will examine the state of the art.

3.2 Twitter

Every social network consists of basic building blocks. The basic building blocks of every social network are: Presence, Sharing, Relationships, Conversations, Groups and Reputation. These blocks are all linked to the users identity [27]. The identity of a user on a network is how they want to reveal themselves. In Twitter the blocks Sharing, Conversations and Reputation are most important. Users share messages through Twitter to interact with each other. Reputation is achieved by the follower count. The more followers, the more influence the user has. Conversations are also important at Twitter, because everyone can react to each other and share their thoughts. Other social networks has their focus on different blocks. Facebook has more focus at relationship and Foursquare has more focus at presence.

3.3 Data of social networks

Social sciences has a long history of surveys for gathering social data. A new upcoming area of social science is computational social science [32, 37]. This area examines data that is produced by users. The data can consists of money transactions, Facebook messages, shopping order data and any other action that is logged. This information can be used to characterize and study a user. Nowadays, analysis of this data is already executed by Google, Yahoo, National Security Agency and many more. Some mayor challenges are ensuring privacy and analyzing the huge amount of data.

Currently, social data is easy accessible using API's for developers [11]. The API's provided by the companies can be used to access the data of the user. The user has to give permission of sharing their information to the application. This is ensured by using the OAuth protocol [9]. The data is usually returned using json [6] and can be accessed using RESTful API's [42].

Every message on Twitter can be shared, favorited or replied on. Kwak et al. [30] shows that the users mostly are in the same timezome and there are some topics that have a small user base, but with a lot of tweets. Furthermore, popular topics are discussed more than specific topics.

3.4 Retrieving additional information

While constructing a dataset all relevant data should be collected. Or when searching on a topic, all relevant documents should be returned. There are some techniques to find additional keywords to improve the query in order to retrieve more relevant documents. This task is called *query expansion.* To determine additional terms there are several methods developed [17]. To overcome the problem of searching only for the exact word, stemming can be applied on the query and document. Stemming will solve the different conjugations of the same word. For instance, the query *work* will be expanded to *working, work, works and worked* or converted to a simple form using a stemmer (e.g. Porter stemming [10]). Another method is expanding a concept [49]. Adding different notations of the query *USA* to *United States, United States of America, the States.*

3.5 Relevance feedback

The resulting documents of the initial query holds additional information. The simplest method of finding additional keywords in a resulting dataset is TF/IDF. Tokenize a text and count each token. Append the most occurring tokens to the initial query. This method is called *relevance feedback*[35]. A form of relevance feedback is that user adjusts the query results by giving feedback to the system, e.g. marking correct results. New results are then received using the adjusted and improved query. A variant of relevance feedback is: pseudo relevance feedback. In this type of feedback there is no explicit feedback from the user required. Feedback is acquired by automatically parsing the results and add some query terms. Such as, always adding the most common word in the top 5 documents. The problem with relevance feedback is that *query shifting* can occur. The results can shift into one direction, which can be the wrong direction. For example, when searching for *jaguar* images, the top results are jaguar cars. The jaguar cars results are used to improve the query. The user, however, wants pictures of a jaguar animal, but instead the user only sees images of jaguar the car. Looking at the topic of a query or a document set is researched by Lavrenko [31]. They are measuring topic similarity. The similarity is used to find additional terms using the new found similar documents. Croft [19] is using language models to introducing some metrics to find specific and clean words. This metric is called clarity. It compares the word distribution and the query distribution.

Another way of identifying keywords of a dataset is by looking at individual tweets. Zhao et al. [52] is extracting key phrases in order to detect the topic of the tweet. They are assuming that each tweet has only one topic. The extracted key phrases consists of multiple specific keywords, which identifies the topic. They are evaluating against 10 extracted topics. To check if a topic was correctly identified, they asked two people to review the topics. A follow up study of Zhao et al. [51] shows that using keyword graphs improves the accuracy. The position of a keyword was found to be important.

Another method for ranking and selecting keywords is called Binary Independent Model (see equation 3.1). Where the terms are considered independent of the relevant and the whole dataset. Where $P(t|R)$ is the probability of term t occurring in the relevant dataset R and $P(t|C)$ is the probability that term t occurs in dataset C.

$$log\frac{P(t|R)(1 - P(t|C))}{P(t|C)(1 - P(t|R))} \tag{3.1}$$

The top ranked documents are parsed and the occurring words or phrases in the documents are ranked using some function. The top additional words are added to the original query to find more relevant documents. The ranking functions can be based on probability, weights or a statistical function, such as Chi-square. To find additional keywords the query log can also be exploited. Regular phrases or keywords can be found and be added to the query. In section 4.1.2 our keyword ranking method is proposed in order to find important keywords.

3.6 Classification of data

A dataset can be labeled or unlabeled [35]. Labeled says that each document is linked to a topic. For example, a document about *Porsche 911* is labeled as *sports car*. A document can have multiple labels and there can exists a hierarchy of labels. Searching or browsing a labeled dataset is easy to use, because the dataset can be filtered at with labels. In order to create a labeled dataset some techniques are proposed. A method is label the document by hand. This is a time consuming task for many documents and a automated approach is required.

A automatic classifier can automatically add labels to a set of documents. There are standardized datasets that are used to compare the performance of different classifiers. *Reuters-21578* [21] is a dataset that is used by many papers to test their classifier performance. The dataset consists of 21578 news articles appeared in 1987 in Reuters news newswire. The articles are classified into 135 categories. The classifier can be tested against this dataset in order to the performance. News articles are long articles and another dataset is needed for short texts.

Short text classification is studied by many researchers [46, 16, 39]. Phan et al. [39] are solving the problem of limited features, by creating a universal dataset that each term in short text can be expanded to. The external dataset consists of news articles and Wikipedia pages. External information is also used by Zelikovitz et al. [50]. They calculate the similarity to some background data, the closest match with another labeled article that has the same similarity to the background data is considered to be the closest match.

A specialized area are Tweets, because Tweets are at most 140 characters long. On average a tweet has 30 characters [2]. Due to the spareness of words/characters short text classification has become challenging. Filtering the tweets can be executed by identifying several classes. This is proposed by Sriram et al. [44]. In this research they have several classes: Sports, News and Chatter. They are combining several hand made features to filter the tweets. Classification of a stream of tweets is done by Dan et al. [20]. They are filtering a stream of tweets related to a TV-show. The goal is to show relevant tweets live on the TV-screen. More domain specific research is done by Prasetyo et al. [41] by looking only at Tweets containing a hashtag of a list. Their goal is to only return domain specific tweets, such as software related tweets.

In Tweets and other social data there is also meta-data. For example, the created date, timezone, username, user language and many more. This information can be used to classify tweets as well. Determining the location of tweet can be done solemnly using tweet-meta data. Li et al. [33] they are determining the location by looking at the context of a tweet and time. Kinsella et al. [28] is examining the possibility to determine a location at city scale.

To automatically adapt, filter and create high quality datasets, frameworks are proposed in order to deliver a standard. These frameworks consists of storing, filtering and parsing data in order to create the correct data for the dataset. This is proposed for by Eugene at al. [22] executed for Yahoo Answers. For finding events, a framework is proposed by Abdelhaq et al. [12].

3.7 Classifiers types

As described by Manning et al. [35] and Kotsiantis et al. [29] there are several different algorithms for classification. The most applied method is Naïve Bayes. The Naïve bayes classifier is using probabilities of words in a dataset in order to classify documents. The classifier is trained on a dataset with several classes. Each term in a document is counted and linked to the label. To calculate whether a new document relates to a class all probabilities of each term in a document are summed. The class with the highest sum is considered to be the best class. There are several extensions on this Bayes model. One variant is the Multinomial Naïve Bayes. This engine can be used when the total dataset is not known [26].

Another used method for text classification is Support Vector Machine (SVM) [15]. SVM can be used in binary classification tasks. Each document is mapped to a vector space. SVM is trained against documents in the vector space. After training the SVM tries to find a line that bisect the classes. The newly documents are also mapped into the same vector space and it can easily be checked whether the document is at a side of the line. That is used to classify the new document. Mostly the vector space cannot be bisected by a single line and there can also other functions be used that bisect the points in the vector space [45]. For information can be found in chapter 5.

There are many other techniques such as k-NN clustering, decision trees and neural networks [35]. These are not considered, because SVM and Bayes should perform reasonable good for our task.

3.8 Current Application and Usages

Live filtering of a stream Tweets is researched by Dan, Fend and Davison [20]. Finding all Tweets of a TV-show is bootstrapped by a set of annotated Tweets that are related to the TV-show. From this initial dataset they are training a classifier which is used to filter the stream of Tweets coming in. All the new relevant Tweets and some additional features are used to improve the classifier. The features consists of some patterns people often use in tweets for a TV-show (e.g. s05e11, season 5 episode 11) and searching for exact TV-show titles.

Finding all relevant tweets about a TV-show is conducted by Ward [47, 48]. The original query of a TV-show is expanded to a longer query in order to access tweets that would otherwise be missed. The method is called Tweet Collect. The additional keywords are ranked using co-occurrence heuristic on keyword pairs. Ward is using unsupervised algorithms, because there is not annotated datase. SVM was found to be imprecise due to the sparse amount of words. To expand the initial query external sources are called. The sources consists of Wikipedia, IMDB and more. On average an F1-measure was determined to be around 85% for manual annotated data and 77% for automatically generated training data.

Plachouras, Stavrakas and Andraou [40] have developed a data collection tool for Twitter. The tool allows researchers to collect tweets about campaigns, events or other tweets. They focus to improve the recall for those events. Starting with tracking a hashtag in order to retrieve the a stream of tweets. They classify a tweet as relevant or missed-relevant (non-relevant). When a tweet is missed it can be either a false positive or a true negative. The recall is calculated by dividing the relevant against he missed-relevant tweets. The recall was around 90% for normal situation. However, by a fast drifting situation as they monitored in the Boston bombings the recall stopped at 44%. This was caused by the shift of keywords after the bombings. A fast adaptive algorithm must be developed in order to solve this problem.

3.9 Conclusion

Additional keywords can be found using existing technologies, such as relevance feedback and automatic query expansion. A relevant dataset can be created using a classifier. Currently, other researchers are focusing on filtering a stream of Tweets for a TV-show.

4

Identifying additional keywords

This chapter is related to research question **Q1**. *How to create a more complete dataset on a topic by identifying additional keywords.* Currently, we have a dataset that is incomplete. For example, the movember campaign. To retrieve more data related to the campaign we propose a method to find additional keywords. Each method establish a score function for a word (t) in the dataset. This $score(t)$ function is used to rank the words in descending order. The word with the highest score is at the top of the list. The different scoring functions are compared and evaluated in section 4.2.

4.1 Method

Firstly, a baseline is constructed. The second method will using probability of a word and the last method will use estimates of the word volume on Twitter.

4.1.1 Baseline with term frequency

A baseline is created for comparison. In order to see the improvements or changes made by the other methods. This method is using individual words. Also called *bag-of-words*. Several steps are required to establish the score. First, the tweet needs to be tokenized. So, each tweet is split into individual words/tokens. Second, each word is counted and stored in a table. When the list is ordered on their word count, the highest occurring terms are on top of the list. The terms that occur the most, are probably words that are highly related to the topic. Unfortunately, there are also terms that are not directly related to the topic. For example, stopwords or common used phrases in a language.

$$score(t) = n_t \tag{4.1}$$

The list is ranked using score function 4.1. Where n_t is the amount of occurrences of term t in the whole dataset. With the ordered list in mind the top k of terms could be used to retrieve more data. The main advantage of this approach, is that is really easy to implements and only the training data needs to be considered.

4.1.2 Using probabilities

The second model is using probability of words. Lets say there are two datasets. One about the *topic* and one *not* about the topic (the not-dataset). The *not-dataset* consists of a hour of Twitter streaming API data. It contains words in 5 common languages (i.e. nl, en, de, fr, se, it). The main concept is based on comparing the probabilities of each word of each dataset. The probability that term t occur in dataset *topic* is compared against the probability that term t occur in dataset *not*.

$$\text{t in topic} = P(t|\text{topic}) \tag{4.2}$$

$$\text{t in not} = P(t|\overline{\text{topic}}) = P(t|\text{not}) \tag{4.3}$$

The ranking of these terms are based on the factor between the *topic* and *not*. For example, the probability of term Movember is 2200 times higher in dataset *movember* than in dataset *not*. This leads to words that are occurring a lot more in the dataset about the topic (see equation 4.4).The top k words are the most notable. These words are related to the *topic* dataset and are used to identify additional keywords.

$$score(t) = \frac{P(t|\text{topic})}{P(t|\overline{\text{topic}})} \tag{4.4}$$

An advantage is that the stopwords are automatically filtered. The probability of a stopwords in *movember* is the same as in *not*. Dividing those two probabilities will return about 1. Only high ranked words are considered as additional keywords. Furthermore, probabilities can easily be calculated using term frequencies. The main disadvantage is needing two datasets. One about the topic and a not-dataset. Calculating the probabilities will take longer due to using two datasets.

4.1.3 Keyword overlap

A method was developed that can predict the amount of tweets fulfilling a query in a year using the Twitter search API. This is later used for ranking additional keywords as well. This method will use a query (q) and with the parameters *until* and *since* the range in a year can be limited. The top 20 results are extracted and the density (expressed in seconds between tweets) is calculated. This number is converted into the amount of Tweets for two weeks. All the results are summed and a estimate of the amount of tweets is generated. It measures the density at 24 moments in a year. Each moment represents 15 days. The density is expressed in the average time, in seconds, between two Tweets. To calculate the amount of Tweets at that moment we use algorithm 1.

Algorithm 1 Pseudo code to estimate the amount of tweets

```
function AMOUNT($q, startDate, endDate$)
    $tweets \leftarrow 0$
    for each Moment from startDate until endDate do
        $tweets \leftarrow tweets + \frac{15*24*60*60}{\text{DENSITY}(q, Moment)}$
    end for
    return $tweets$;
end function
function DENSITY($q, Moment$)
    $results \leftarrow \text{GETTWEETS}(q, Moment)$
    $seconds \leftarrow 0$
    $lastDate \leftarrow null$
    for tweet in results do
        if $lastDate \neq null$ then
            $seconds \leftarrow seconds + (tweet.createdAt - lastDate)$
        end if
        $lastDate \leftarrow tweet.createdAt$
    end for
    return $\frac{seconds}{results.length}$
end function
```

Twitter does support advanced search, this can be exploited to find a ratio between two words. In figure 4.1.3 we have #Movember and #Moustache. *Amount* predicts that #movember have 652.360 tweets in the year 2013/2014. Table 4.1 lists all the combinations and estimations of the amount of tweets.

To determine the score of each word the overlap is calculated between the original keyword or search query q and the new proposed word. This function is using algorithm 1 for determining the amount of tweets.

$$score(t) = \frac{\text{Amount(} q \text{ AND } t \text{)}}{\text{Amount(} t \text{)}} \tag{4.5}$$

$$\frac{\text{Amount(\#Movember AND \#Moustache)}}{\text{Amount(\#Moustache)}} \tag{4.6}$$

As an example #Moustache and #Movember are used (equation 4.6). When score is 1,

Combination	Number of Tweets
#Movember	652.360
#Moustache	88910
#Moustache AND #Movember	8267
#Moustache AND ¬#Movember	80809
¬#Moustache AND #Movember	650335

Table 4.1: Estimates of #Movember and #Moustache combinations

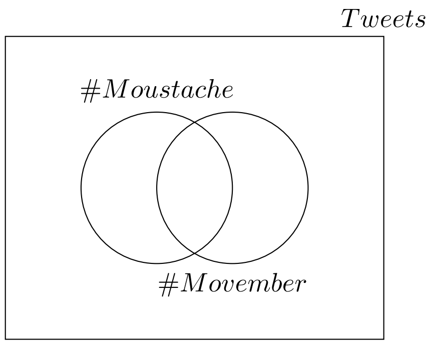

Figure 4.1: Venn-diagram for #Movember and #Moustache

the word t is not used outside the original topic. When the overlap is 0 the word does not provide additional documents related to the topic. An indication if a word is *good* or not, can be established by the relationship to a known term. In figure 4.1.3 the relationship can be expressed in two circles. Assuming that any Tweet containing the word #Movember is about the campaign movember (later more on this assumption). There are Tweets that has #Movember and #Moustache, those are related to movember. There are, however, tweets that does not have the #Movember, but has the word #Moustache. The amount of overlap indicates a relationship between those two terms.

$$\begin{aligned} score(t) &= \frac{\text{Amount(} q \text{ AND } t \text{)}}{\text{Amount(} t \text{)}} && (4.7)\\ &= \frac{P(qt)}{P(t)} && (4.8)\\ &= \frac{P(q|t)P(t)}{P(t)} && (4.9)\\ &= \frac{P(q|t)}{1} && (4.10)\\ &= P(q|t) && (4.11)\end{aligned}$$

The function Amount estimates the amount of Tweets in a timespan. This can easily be converted into probabilities by dividing that count by a estimate of the total volume of Tweets on Twitter. In equation 4.8 the score function is converted into a probability function. The probability space is expressed in the total volume of Tweets in a year. By applying Kolmogorov definition in 4.9 the formula is converted into a conditional probability. This can be cleaned to 4.11. Showing that the score function is expressing the probability that q happens while t.

4.2 Validation

Three methods are tested: the baseline (TermFreq), using probabilities (ProbFact) and overlap (Overlap).

4.2.1 Evaluation

There are several methods for evaluations. An automated evaluation approach will be used and an evaluation by hand is performed. To check if the list created by a method is satisfying or not, the list of keywords will be judged by hand using the following rules:

Rule 1 word t is not a starting with a sign, because interactions between users are platform specific.

Rule 2 word t is not a link

Rule 3 word t is a word and not a combination of non-alphabetical characters.

Rule 4 word t is related to the original topic. A word is related when the word is known to be related by the person who selects the words.

The accuracy of the evaluation by hand is calculated by dividing the amount of good picks with the total amount of words.

$$Accuracy = \frac{\text{count related t}}{\text{count t}} * 100 \tag{4.12}$$

4.3 Data

To receive enough information for our evaluation, events are used as a source. For example, the MotoGP in Spain. Some initial data is received using the Twitter search API, using the hashtag of MotoGP in Spain(i.e. *#SpanishGP*). Previously described methods are applied, in order to find additional keywords. The top k of those keywords are used by the Twitter Streaming API. After each event the received Tweets will be used, in our evaluation to see whether the keywords delivered additional Tweets about the event. This will be performed against several events, such as USOpen, Motor races and other events. The Twitter Stream API only returns 1% of the total stream. This part of the stream can be considered as a representation of the whole stream [38]. In section 5 the classifier for this task is created. In section 6 the results of the additional keywords is evaluated in an integrated approach.

4.4 Results

Topic	**Query**
Frech MotoGP	#frenchGP OR #MotoGP
Giro d'Italia	#giro
TV-show Game of Thrones	#got
Movember campaign	#movember

Table 4.2: Used Events

We have gathered keywords for several events and campaigns. In table 4.2 the events are listed. For each event additional keywords were identified using all three methods. For the event French MotoGP the keywords are explained and calculated as an example. The remaining events are listed in table 4.7. For each method and event we have determined an accuracy by hand. As described in previous section 4.2. In chapter 6 the actual results, using the classifier in chapter 5, are shown for each event.

4.4.1 Example *FrenchGP*

We will start with method *TermFreq*. The returning list is filtered using a stopword filter and the top terms are determined by their count. Table 4.3 shows the top 15 keywords using tweet frequency. The terms *#frenchgp* and *#motogp* are well determined. Meanwhile the term *from* is very general. The accuracy for this top 15 is *40%*.

Keyword	Frequency	Relevant
#frenchgp	668	1
#motogp	525	1
mans	200	1
pole	172	
#lemans	141	1
tomorrow	127	
race	90	
position	88	
from	80	
today	75	
rossi	71	1
el	63	
best	61	
watch	59	
marquez	59	1

Table 4.3: FrenchGP keywords using frequency

Probability factor When applying our second method which is based on a factor between the FrenchGP data and normal data (called ProbFact). As shown in table 4.4 the probability of *#motogp* is 5147 times higher in dataset *FrenchGP* than in the normal dataset. The keywords *#motogp, #frenchgp, #lemans, rossi, marquez and #howlowcanhego* are strongly related to FrenchGP. The overall accuracy for this event is 47%. Because 8 keywords are not directly related to FrenchGP and 7 of 15 are.

Overlap percentage *Overlap* will use the top 75 keywords of *ProbFact* as input for the method overlap. The overlap tool is very intensive for the servers of Twitter, so we did choose not to rank all the keywords. But, only the top 75 keywords. In table 4.5 the keywords ranking using overlap are listed. Looking at the accuracy of the top 10 is 100%, for top 15 it is: 90%. The hashtag #lemans is related to FrenchGP, however not to all MotoGP, because it was the race track. For another MotoGP event this hashtag will be invalid.

The overlap percentage is calculated using the *Density* function. It should be noted that the standard deviation of the Density functions accuracy is around 31%. So, the predictions of the tool can deviate a lot from the real results. However, this information is still useful, because on average the tool is correct.

To see the accuracy of other events this method is also applied on the Movember campaign.

Keyword	Factor	Relevant
#motogp	5147	1
#frenchgp	3274	1
pole	1686	
#lemans	1382	1
mans	980	
rossi	696	1
marquez	578	1
demonstrates	558	
row	549	
slomo	539	
#howlowcanhego	509	1
prepares	509	
morning”,	470	
motogp	460	1
saturday’s	460	

Table 4.4: Game of Thrones keywords using ProbFact

Keyword	#frenchgp and Keyword	Keyword	Overlap %	Relevant
#frenchgp	5538	5538	100.00	1
#motogp	38261	38261	100.00	1
#mig16	29	29	100.00	1
#howlowcanhego	32400	32400	100.00	1
#tuttoacceso	776	1068	72.66	1
#skyracingteam	29	41	70.73	1
#moto2	4988	7108	70.17	1
#monstergirls	431	1510	28.54	1
#lemans	2497	13766	18.14	1
motogp	39179	286305	13.68	1
asciutto	187	2199	8.50	1
lemans	2138	33023	6.47	1
pedrosa	4422	257875	1.71	1
quali	975	74244	1.31	
qualified	3063	265714	1.15	

Table 4.5: FrenchGP keywords using Overlap

Table 4.6 shows the keywords for #movember and each method.

In table 4.7 all events are compared against each method and the overall result is quite good for the method *Overlap* and even for *ProbFact*. These accuracies are determined by hand. In the chapter 6 each top 10 of keyword is evaluated against the real (received) data. A keyword can be classified by hand as relevant, however the question still stands if the keyword adds relevant tweets to the dataset.

In figure 4.2 the degradation of each method is shown. The accuracy is calculated for every top *k* keywords. For example, at position 10 the accuracy is determined by the percentage of

Overlap	**DocFreq**	**ProbFact**
#movember	movember	movember
"movember"	#movember	#movember
movember	join	moustache
#menshealth	team	shave
#mustache	grow	november
moustaches	good	beard
moustache	moustache	enlisted
#mo	month	mustache
beards	shave	tash
mustache	start	grow
stache	health	mens
tash	like	signed
beard	donate	participating
mo	time	month
growing	growing	ive
86%	26%	53%

Table 4.6: Movember and the different methods

Event	**Overlap**	**ProbFact**	**TermFreq**
#giro	80%	86%	60%
#frenchGP	100%	47%	40%
#movember	86%	53%	26%
#got	80%	60%	26%
Average	86.6%	55%	38%

Table 4.7: Events and methods compared

relevant words of the top 10. At position 40, the accuracy is determined by the percentage of relevant words in the top 40. The method *Overlap* does perform the best, followed by *ProbFact*. *TermFreq* is not reliable because only 10% of the terms in the top 50 are relevant. For *Overlap* this is around 50%. Looking at the top 10, the *ProbFact* accuracy is around 75% which is still useful for our application.

4.5 Conclusion and discussion

The discussed methods were evaluated by hand and as a result the method *Overlap* returns the best additional keywords. *TermFreq* does not have a proper filter and the accuracy is low. The probability method is promising due to the fact that is very easy to apply and does not require slow calls to Twitter.

TermFreq In TermFreq each term is counted. Tweets using these terms can contain additional keywords, but they are not directly connected to the topic. The terms can relate to the natural language of sub-language of Twitter. The stopword filter does improve the list, however filtering

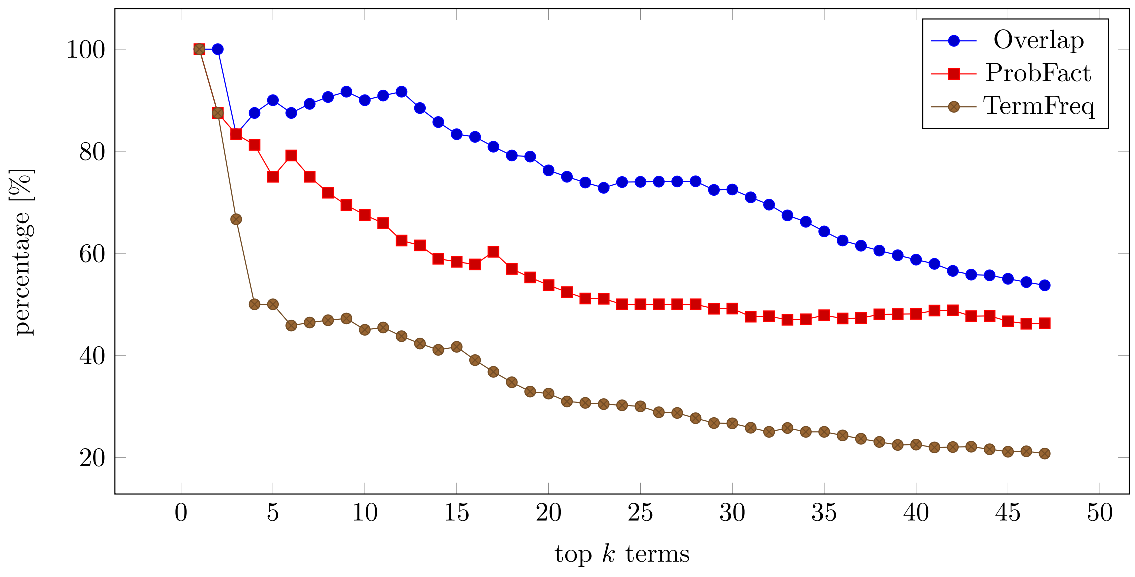

Figure 4.2: Top K keywords and accuracy

normal (not stopword) words is not covered. Overall can be said that some words adds additional information, but the accuracy of the top 15 using document frequency is on average 38%. So, most of the words in the top 15 are not relevant and should not be used.

ProbFact Finding notable words using probability was found to be successful. Comparing the probability of a term to the probability in the normal language, keywords that are notable in a topic were found. These method can also be applied without a stopword filter, because the probability of common words are filtered out, because the probability is the same in the normal and original dataset. A downside is that the occurrence of a word is important. A vocabulary mismatch is also a problem. There are words that are in topic, but not in the not-dataset. This is solved by applying La Place smoothing [5], keeping the formula correct. There are also terms not used a lot, but are very related to the original topic. Furthermore, the probability of a term needs to be calculated of two datasets, needing more memory and processing power. A upside is that there is no need of external sources and the terms can be calculated very easily. Overall accuracy is around 55%.

Overlap The keywords that are listed using *Overlap* returns the best keywords. On average with 86.6% accuracy. The main downside of this method is that it utilize Twitter search a lot. For every keyword the Twitter search page is called 10 times. The total duration per keyword is around 5 seconds. Thus, constructing the list is very slow. Determining the top 15 list takes about 15 minutes. However, the top 15 list contains a lot of relevant keywords. The overlap does not guarantee that the lists adds additional tweets, because the higher the overlap the more probable it is that a keyword is related to the topic, but it will return less Tweets. And a low overlap does not mean that a keyword is not related to the topic. A keyword can be related to the topic, but not used frequently compared to the original search topic. Detecting those keywords is still a challenge, because the keyword is not used a lot and is highly relevant.

The real accuracies are compared to our initial accuracies determined by hand. And *Overlap* is the most promising, followed closely by *ProbFact* due to its simplicity and speed.

5

Filtering for relevant Tweets

This chapter is related to research question **Q2**. *How to create a relevant dataset on a topic by filtering the dataset?* Lets assume, there is a dataset tweets with the keyword *cancer*. We are only interested in tweets related to the illness cancer. There are some Tweets in the dataset that are not about the illness cancer. To filter out the irrelevant tweets, a classifier is created that is able to handle short messages and can successfully filter out the irrelevant tweets by looking at the Tweet text. A Naïve Bayes classifier and a Support Vector Machine classifier are used as classifiers.

Short messages are more difficult due to sparse features. In long texts there are a lot of features, for instance all the words. However, in short texts (in our case up to 140 characters) does not contains many features. On average a tweet is only 28 characters long [2]. This causes that some features/words are very important for the classifier. Proper handling these features is required, in order to distinguish relevant and irrelevant tweets.

5.1 Method

There is not a standard dataset for short text classification. In order to annotate the tweets we assume that when a Tweet contains a hashtag H the tweet is about the topic related to that hashtag. For example, for the hashtag #Movember it is assumed that the tweet containing #Movember is related to the movember campaign (see Definition 1). This definition will be validated in section 5.3.

Definition 1 *When a tweet contains hashtag H, the tweet is about the topic of H*

5.1.1 Classifiers

Naïve Bayes First of all Naïve Bayes is used as a classifier. Naïve Bayes is used a lot in research [35] due to it simplicity and combining features. It can easily handle multiple features for determining the class of text. The Bayes classifier can easily be trained and classification of a new document can be calculated quickly.

In Naïve Bayes a string of characters is tokenized into words. Tweet text is not tokenized on # and . These characters are used for a hashtag respectively a reply. For each Tweet the words

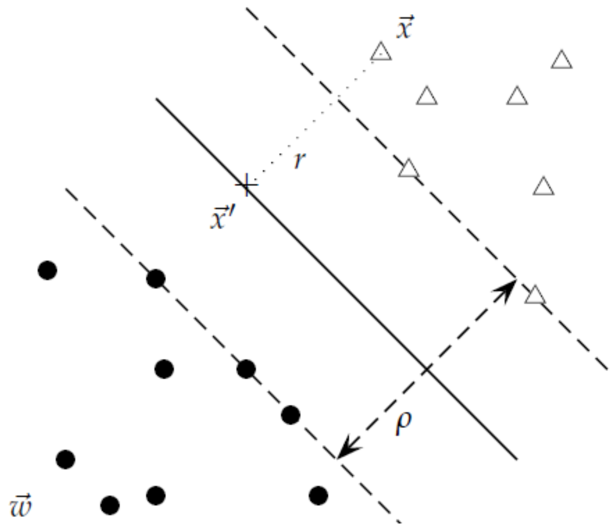

Figure 5.1: SVM Vector Space [35]

are counted into their corresponding classes. After training the probabilities of each word in a class is calculated. The class with the highest score is found to be the most probable class. The score is calculated with equation 5.1. Where $P(t_k|c)$ is the probability of term t_k in a document in class c. The probability of occurring of class c is $P(c)$. To find the best class we use the maximum value of all the classes ($c \in C$) (see equation 5.2). The best class for a tweet (d in equation 5.2) is determined by selecting the class with the highest value.

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{5.1}$$

$$class = \arg\max_{c \in C} P(c|d) \tag{5.2}$$

Support Vector Machines Support Vector Machines do not use probabilities, but a vector space [25, 35]. Each tweet is tokenized and mapped into a vector space (see figure 5.1). Each training point in the vector space is related to a certain class. After training a kernel function has to be found. A kernel function is a function of a ”line” that can bisect all the points in the vector space (black line in figure 5.1). Finding this kernel function can be hard, because there is never a perfect vector space which has a clear separation of classes. When a new document needs to be classified the document is mapped into the vector space. If this document is above or below the kernel function the document is either classified into the one or the other class.

The implementation of Weka [24] is used for the Bayes classifier (NaiveBayesMultinomial Java Classifier) and the SVM classifier using LibSVM [18]. The Multinomial Naive Bayes is using standard settings. The SVM classifier is configured using the following parameters. It uses a two degree polynomial kernel.

```
SVMType: C-SVC
coef0: 0
```

```
cost: 1
degree: 2
eps: 0.001
gamme: 0
kernelType: polynomial
nu: 0.5
```

To find the overall accuracy, precision and recall, 10-fold-cross validation is used for the Reuters dataset and a separate training and test set is created for the tweet classification.

5.2 Data

To test whether our classifiers are behaving correctly, they will be compared to Reuters-21578 [1] set. The Reuters-21578 dataset is a set of news articles about several topics. To see the impact of short messages the previously crawled Tweets from chapter 4 will be used. The hashtag assumption is used to annotate the Tweets. The training set is created using all the tweets containing a certain hashtag. In our example we will use #frenchGP. Thus, all the tweets containing the hashtag #frenchGP will be used as training data. The hashtag itself is removed, preventing that the classifier only relies on the hashtag itself and not on other features. All the tweets that are retrieved using the additional keywords, will be used in our evaluation. The keywords that were found to be relevant of chapter 4 are used as a testset. For example, a Tweet that not contains the original hashtag #frenchGP, but a related hashtag will be annotated as relevant in the test set.

5.3 Evaluation

Multi class classification vs binary classification This research has a binary classification problem. To convert Reuters into a binary classification problem. Training set will be constructed for each topic and mark all other topics and tweets to the class *not relevant.* The classifier should be able to select all related tweets.

5.4 Results

The Reuters dataset is used to test the performance of the classifier on normal texts. In order to compare the two datasets the Reuters dataset is converted to a binary classification. For 10 categories the classifier is trained and tested. These categories are: *Earn, Acquisition, Money-fx, Grain, Crude, Trade, Interest, Ship, Wheat and Corn.* These 10 categories are the biggest categories and are used in other classification research [25, 34]. The results of the different classifiers for the Reuters dataset can be found in table 5.1. The precision is only for the class itself. For the class *corn* the precision was 0.89, saying that in 89% of the classified corn documents were real corn documents. The recall for corn is 0.47, saying that only 47% of the corn documents were found. On average the precision for the SVM classifier is really good. Compared to the literature these results about the same [25]. The Bayes classifier is performing like expected. The average precision of Bayes is around 75%.

For classifying Tweets a high precision classifier is required, because only relevant Tweets are allowed in the dataset. In the next section the hashtag assumption will be checked and the accuracy for short messages will be determined.

	SVM			**Bayes**		
Class	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
acq	0.937	0.851	0.891	0.861	0.871	0.865
corn	0.898	0.475	0.621	0.444	0.083	0.139
crude	0.893	0.649	0.752	0.835	0.613	0.707
earn	0.985	0.915	0.948	0.959	0.843	0.897
grain	0.901	0.658	0.760	0.785	0.664	0.719
interest	0.813	0.580	0.677	0.749	0.486	0.589
money-fx	0.867	0.669	0.755	0.744	0.713	0.727
ship	0.902	0.514	0.654	0.881	0.542	0.670
trade	0.845	0.649	0.734	0.602	0.625	0.613
wheat	0.860	0.660	0.746	0.689	0.215	0.328
avg.	0.890	0.662	0.754	0.755	0.566	0.625

Table 5.1: SVM and Bayes on Reuters-21578 dataset for each class

5.4.1 Hashtag assumption

The hashtag assumption (described in definition 1) is used for annotating the tweets. To check of the assumption is correct, 100 tweets of each events, containing the hashtag, are randomly selected. These tweets are checked by hand, if a tweet contains the hashtag and is about the topic of that hashtag. In table 5.2 the accuracies of each event is listed. On average the accuracy is 94.5%. The assumption is almost correct and is usable in this research.

Event/Campaign	**Accuracy**
#movember	92%
#giro	99%
#BigBangTheory	97%
#Arrow	100%
#GoT	80%
#FrenchGP	99%
avg.	94.5%

Table 5.2: Accuracy of Hashtag assumption

5.4.2 Classification

The classifiers are evaluated in two phases. The first phase is to see if the classifier is able to detect relevant tweets. These tweets are obtained using the streaming API listening to several languages. The results are shown in table 5.3. The second phase is evaluated against all events to see the effect for different classifiers. For each event the precision, recall and F1-measure are shown for the relevant tweets. In table 5.4 the precision and recall are shown for the events.

First phase Both classifiers are able to detect the relevant tweets against the totally not relevant dataset (see table 5.3). This method uses 10-cross-validation for the overall precision and accuracy. The SVM classifier is able to distinguish almost all relevant Tweets against not

		SVM			**Bayes**		
Event		**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
FrenchGP	With #frenchgp	0.997	0.981	0.989	0.94	0.993	0.965
FrenchGP	Without	0.995	0.981	0.988	0.94	0.993	0.965
Arrow	With #arrow	0.996	0.978	0.989	0.937	0.988	0.962
Arrow	Without	0.995	0.983	0.989	0.937	0.988	0.962
Movember	With #movember	0.989	0.963	0.976	0.929	0.981	0.954
Movember	Without	0.952	0.963	0.985	0.929	0.981	0.954
Game of Thrones	With #got	0.998	0.890	0.941	0.951	0.914	0.932
Game of Thrones	Without	0.998	0.924	0.960	0.951	0.914	0.932
Giro d'Italia	With #giro	0.999	0.932	0.964	0.931	0.984	0.956
Giro d'Italia	Without	0.996	0.934	0.964	0.931	0.984	0.956
avg.	With	0.996	0.949	0.971	0.938	0.972	0.954
avg.	Without	0.987	0.957	0.972	0.938	0.972	0.954

Table 5.3: Results classifier against normal tweets

relevant Tweets. The Bayes classifier is also performing well. On average Bayes can detect 94% of the cases. Both classifiers have a high recall above around 95%. The impact of the original hashtag is small. The classifiers are performing almost the same with or without the hashtag.

Second phase The second phase consists of evaluating the classifiers against their own topic. For Giro, Game of Thrones and FrechGP there are not many relevant tweets. This is because of the limitations of the streaming API. In the previous chapter, we have submitted around 30 keywords, where some were relevant and some were just generating noise. Thus, the amount of relevant tweets is not very high. For TV-show the keywords were selected with ProbFact method and only 10 keywords were used. Leading to a much bigger training and test set.

Looking at table 5.4 the training and test sets are explained in column 2. For Game of Thrones 316 of the 15316 Tweets contains the hashtag #got and 15000 Tweets were not relevant Tweets. The same applies for the test set where 1057 of 19057 Tweets were relevant Tweets and 18000 Tweets were not relevant Tweets.

The SVM classifier did not perform well when the original hashtag was included. The average F1 for SVM with training on the hashtag was 0.004. However, Bayes did perform really well when the hashtag is used for training. The F1 ended up at 0.682 with an average precision of 0.847 and recall of 0.700. When the hashtag was not included in the training data SVM was able to return some scores. For Arrow it was able to detect relevant arrow Tweets with an precision of 0.914 with a recall of 0.321. Looking at the results, a Bayes classifier should be used and training the classifier should be done using the Tweet without removing the original hashtag.

Table 5.3 suggests that removing the original hashtag has no impact. However, table 5.4 shows that it has some effect. This effect is probably caused by the training algorithm of SVM. In SVM the vector space is reduced/cleaned by removing vectors that are not supporting. For instance, are not at the edge of a group. When a hashtag is included in the training Tweets, SVM detects that all the Tweets has the hashtag and it will remember that the hashtag is very important for the training data. In this experiment the test documents are not containing the

			SVM			**Bayes**		
		Train with/without hashtag	Precision	Recall	F1	Precision	Recall	F1
Giro	train: 20410/5410 test: 23990/5990	With	0.667	0.001	0.003	0.628	0.829	0.714
#giro		Without	0.704	0.395	0.506	0.572	0.522	0.546
Game of Thrones	train: 15316/316 test: 19057/1057	With	0.059	0.001	0.002	0.922	0.090	0.164
#got		Without	0.448	0.028	0.053	0.727	0.015	0.030
FrenchGp	train: 46599/31599 test: 54749/36749	With	1.000	0.005	0.009	0.876	0.927	0.901
#frenchgp		Without	0.801	0.388	0.522	0.874	0.576	0.694
Arrow	train: 117638/102638 test: 146962/128962	With	0.000	0.000	0.000	0.963	0.955	0.948
#arrow		Without	0.914	0.321	0.475	0.957	0.485	0.644
avg.		With	0.432	0.002	0.004	**0.847**	**0.700**	**0.682**
		Without	0.717	0.283	0.389	0.783	0.400	0.479

Table 5.4: Precision and recall per event. For train and test: *total Tweet count/relevant Tweet count*

hashtag, consequently the SVM classifier does not have enough information, because it only relied on the hashtag.

5.5 Conclusion and discussion

Two different classifiers were tested. The Naïve Bayes classifier worked well on Tweets. It was predictable and easy to use. With an average accuracy of 85% it is usable in this research program to achieve the main goal. The SVM classifier was also easy to use, it relies on a few support vectors it should not be used. In the first task the SVM outperforms Bayes easily, because there was a clear distinction between the classes in the vector space.

The method for in the second task is biased, because Tweets are used that contains relevant keywords and it was tested against Tweets that were totally not relevant. There are cases that relevant keywords are used in a Tweet, but the Tweet is not relevant. Those Tweets are important to be filtered out. When this method is added as a part to achieve the main goal, this concern is not relevant anymore, because the selected keywords are biased anyway.

The errors made by the classifiers consists mainly of Tweets that are very short or in a foreign and unknown language for the classifier. There are not enough features available to correctly classify the Tweets.

Concluding, the Bayes classifier will be used for post filtering the received tweets using the streaming API. Our previously described method can effectively filter the dataset in order to create a relevant dataset.

6

Retrieve all related tweets

In chapter 4 several methods are proposed in order to identify additional keywords. In chapter 5 a classifier is proposed, which is able to handle short messages. This section related to research question **Q3**. *How to combine filtering and identifying additional keywords in such a way that it will improve the dataset?*

6.1 Data retrieval

The combination of additional keywords with a reliable classifier, will produce more related tweets about a certain topic. An initial query is required to bootstrap the application. A query can be anything of some keywords or a advanced query on Twitter. The application can also be bootstrapped by a carefully selected dataset of tweets or using a query on Twitter that will look at the past 2500 Tweets matching the search query. Twitter search supports more advanced queries, such as *#movember AND november* or *"mark rutte"*. Most times a specific hashtag is enough. For instance, *#movember*.

6.2 Method

After receiving all data using the keywords the data contains probably noise. Adding more keywords, more noise is added to the dataset, because each keyword do not produce only relevant Tweets. All irrelevant tweets should be discarded. This can be done using the classifier as described in chapter 5. This classifier is trained on all tweets containing the initial query (e.g. #movember). For each additional keyword the amount of relevant Tweets is counted. The total amount of relevant Tweets in the dataset are also counted.

6.2.1 Final application

To fulfill our main goal, a application needs to created that automatically handles all steps and delivers a complete and relevant dataset. A application was developed using several components. In figure 6.1 the general outline is demonstrated. The program is initiated using a hashtag, that identifies the campaign or event. Additional keywords are found and ranked. *Overlap* is used for identifying additional keywords (see chapter 6.3). The best 10 additional keywords are used as keywords for the Twitter streaming API. The streaming API pushes the received tweets into a queue. Each component can have a input and/or output queue. The queue can be read by the components that are subscribed to that queue. This pattern is called a message queue [8]. This

Topic	**Query**
FrenchGP MotoGP	#FrenchGP OR MotoGP
Giro d'Italia	#giro
TV-show Arrow	#arrow
TV-show Game of Thrones	#got

Table 6.1: Data received

allows components to work simultaneously and parallel on the same queue. Each components have their own concern (e.g. separation of concerns). For instance, outputting tweets into a queue. The next component is the classifier. It takes the output stream of the streaming API and using the training data, it will classify a tweet into a class. When a Tweet is relevant it will be added to the *match* queue. And when a tweet is not relevant to the *no-match* queue. Both queues are read by a writer component. This component takes a output stream and writes the Tweets of the queue into a file. The classifier adaptive update the model using the hashtag assumption. When a Tweet is received containing the original hashtag, this Tweet is used to update the classification model of the classifier. The impact of this adaptive classifier was not measured, due to time limitations of this research. This final application fulfills our goal to create a complete and relevant dataset on a certain topic. In the next section the application is evaluated.

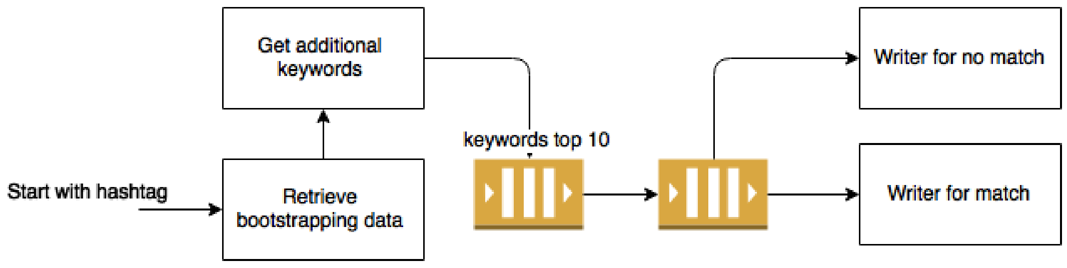

Figure 6.1: Program flow for the final application

6.3 Evaluation

. First of all, the data retrieved using additional keywords is evaluated. In table 6.1 each event is listed. At the beginning additional keywords are identified using the tweet history of 3000 tweets matching the query. For each event the additional keywords will be determined. And for each keyword in that event the amount of newly relevant added tweets will be counted.

Evaluating our final application is executing by analyzing the resulting tweets into their classes. The overall accuracy is determined by sampling 200 tweets of the produced relevant dataset and check by hand if a tweet was relevant. To find the amount of error, the *not relevant* dataset is also analyzed by hand. Ideally, the matching dataset contains less errors. However, the *not relevant* dataset can contain a lot of relevant tweets. That is not a problem, because less noise is preferred over more relevant data.

6.4 Results

In chapter 4 additional keywords were identified. In chapter 5 we are able to filter the tweets to the relevant tweets. In this section an evaluation of the combination is performed. Each additional keyword is evaluated for each method and in this chapter the overall precision and recall of each event is established.

6.4.1 Evaluation of additional keywords

Using all three additional keyword methods the top 10 of them were used to retrieve a dataset. For each event and for each keyword the precision is determined. There are two metrics: the amount of additional keywords and the accuracy of the tweets using that keywords. The listed precision is determined by the classifier of our previous chapter.

Lets say, that a keyword is found to be relevant, when the classifier classifies more than 80% of the Tweets of keyword as relevant. With this threshold in mind, each additional keyword is found to be relevant or not.

TermFreq		**Useful**	**ProbFact**		**Useful**	**Overlap**		**Useful**
#motogp	0.964	1	#motogp	0.964	1	#forzaducati	1.000	1
#frenchgp	0.997	1	#frenchgp	0.997	1	#frenchgp	0.997	1
#q2	1.000	1	#q2	1.000	1	#motogp	0.964	1
pole	0.876	1	pole	0.876	1	#fp4	0.929	1
#lemans	0.960	1	#lemans	0.960	1	#q1	0.662	0
#q1	0.662	0	#q1	0.662	0	crutchlow	0.940	1
q2	0.690	0	q2	0.690	0	#lemans	0.960	1
y	0.087	0	q1	0.637	0	#q2	1.000	1
mans	0.753	0	petrucci	0.506	0	petrucci	0.506	0
			qualifying	0.943	1	motogp	0.958	1
avg. and sum	0.776	5		0.823	6		0.891	8

Table 6.2: Additional keywords accuracy for each method for event FrenchGP

In table 6.2 the accuracy of each keyword is listed. An threshold of 0.8 is applied to determine if a keyword is useful or not (see equation 6.1). For a threshold 0.8 is chosen, because it says that more than 80% of the Tweets using that keyword were found to be relevant. The method *TermFreq* returns 5 useful keywords and an average accuracy over all keywords of 77%. This is not bad for a simple word counter. The method *ProbFact* performs a little bit better. Six new keywords were found to be useful and with an average accuracy of 82%. *Overlap* has found 8 useful additional keywords and with an average accuracy of 89%, this method turns out to be the best performing method in the case of FrenchGP. The same tests are executed for Game of Thrones and Giro d'Italia. The comparison of each method per event is found in table 6.3. For each event *Overlap* performs the best. And *TermFreq* the worst.

For Giro d'Italia the increase in accuracy for each method is substantial. However, for *Overlap* only three keywords have reached the usefulness threshold, while the overall accuracy did increase to 65%. For Game of Thrones no useful additional keywords were found, this is caused by the fact that the used hashtag #got is ambiguous. Some boy band in Korea is using the hashtag as well, producing a lot of irrelevant Tweets and keywords. The retrieved training data was not sufficient for establishing useful additional keywords. Fortunately, the average accuracy is still the highest for *Overlap*. This evidence shows that the method *Overlap* is performing the best of all proposed methods. For FrenchGP and Giro d'Italia the dataset size was increased by 16%. The additional keywords for Game of Thrones are different in chapter 4, because the tests were executed at different times.

The same results were found in chapter 4. *Overlap* did perform the best for the top 10 of keywords. However, in this real life situation less additional keywords were found. Probably caused by the fact that the streaming tests were executed at a different time.

$$useful(p) = \begin{cases} yes & \text{if } p > 0.8 \\ no & \text{otherwise} \end{cases} \tag{6.1}$$

		FrenchGP	**Giro d'Italia**	**Game of Thrones**
	using orig keyword	16326	2138	1182
DocFreq	**avg. Accuracy**	0.776	0.202	0.001
	useful keywords	5	1	0
	added	26107 (+160%)	0 (+0%)	0 (+0%)
Prio	**avg. Accuracy**	0.823	0.512	0.006
	useful keywords	6	2	0
	added	26107 (+160%)	330 (+15%)	0
Overlap	**avg. Accuracy**	0.891	0.649	0.010
	useful keywords	8	3	0
	added	3310 (+20%)	340 (+15.5%)	0 (+0%)

Table 6.3: Overview of each method per event

6.4.2 Overall result

Our final application was tested on three events. A TV-show called Rookie Blue, about a rookie police women. The USOpen of golf. One the biggest golf tournaments. And a formula one race in Austria called the AustrianGP. While running the application statistics about the Tweets and classifier are collected. For each keyword, the number of tweets in a class were counted.

In table 6.4 the results for the TV-show are listed. It demonstrates that some keywords have a lot of noise. For instance, the term *rookie* has 20325 document classified as relevant, but also 13822 as irrelevant. These numbers shows that the term *rookie* are not strongly related to the TV-show. Using these numbers the dataset can be filtered. When the ratio between the classes *match* and *no match* is above 0.2 calculated using $\frac{no-match}{match}$ the keywords adds to much noise and all tweets containing that keyword are discarded. For each event, the accuracy before and after the filter are shown. In table 6.4 the keywords *rookie, gail and traci* have a lot of tweets classified as not relevant. The ratio between the classes is for each of these words above 0.2. The accuracy before filtering was 1%. After filtering (e.g. removing tweets using rookie, gail and traci) the accuracy increases to 71%.

Now looking at the dataset of #USOpen. In table 6.5 the results are shown. In total 179000 tweets were classified as relevant and 12000 as not relevant. Looking at the matching dataset,

Keyword	match	not	not/match
#rbparty	1483	0	0.000
et/pt	608	29	0.048
#renewrookieblue	29	0	0.000
#renewtheblue	21	0	0.000
#nothingwillbethesame	16	0	0.000
traci	1801	399	0.222
#mcswarek	151	0	0.000
#rookieblue	2465	9	0.004
gail	2970	667	0.225
rookie	20325	13822	0.680

Table 6.4: Statistics for each keywords of TV-show Rookie blue

the keywords *#usopen, #tigerwoods, #chambersbay* are the most relevant. The keywords are followed by *dustin, chambers, leaderbord and #royalascot.* The keywords leaderbord and #royalascot are producing a lot of noise. The hashtag #royalascot is about horse racing and was found in the training set, because the stream started before the event had started. Before the event there were a lot of gambling/bidding tweets about USOpen en Royalascot. For example: MORNING ALL! LUCKY FRIDAY?? #USOPEN #ROYALASCOT. TODAY'S OFFERS COMING UP SOON.... #royalascot is not related to Usopen, however due to the training data it was added to our additional keywords. The words *bubba* and *fowler* are both related to players at the USOpen. However, these words are also used in other contexts. The accuracy before filtering was around 62% and after filtering (removing words with a ratio above 0.2) it wend up to 67%. However, when all Tweets are removed containing #royalascot the accuracy was around 90%.

Keyword	match	mot	not/match
#usopen2015	20478	524	0.026
#tigerwoods	3193	41	0.013
dustin	14943	2270	0.152
chambers	38906	1635	0.042
#royalascot	16213	878	0.054
#chambersbay	8313	93	0.011
leaderboard	14424	712	0.049
#usopen	51092	469	0.009
bubba	6318	3704	0.586
fowler	5135	1708	0.333

Table 6.5: USOpen golf tournament keywords and statistics

Applying *Overlap* on AustrianGP did not return great additional keywords. Table 6.6 shows that the keywords *#redseason, #fp3, #f1, #austriangp and #seb5* are useful. This was established using the threshold of 0.2. For the words *994* and *haring* the *not relevant* class is bigger than the *relevant* class. Which is a shame, because a keyword were found to be relevant by our additional keywords method. The precision of the dataset using all keywords was 52%. Applying the filter the accuracy goes up to 92%. After applying the filter, the amount of Tweets increased with more than 95% Tweets.

Keyword	**match**	**mot**	**not/match**
#redseason	6559	155	0.0236
drivers	49322	22943	0.4652
994	1373	1526	1.1114
spielberg	12429	5036	0.4052
#fp3	3700	48	0.0130
haring	507	652	1.2860
#f1	113787	16215	0.1425
#austriangp	135685	6365	0.0469
177	8164	4569	0.5597
#seb5	5213	162	0.0311

Table 6.6: Statistics of each keyword for event AustrianGP

All scores combined All results are combined in table 6.7. Filtering did increase the accuracy. Our tool did manage to add on average 105% tweets with an average precision of 76%. Concluding that recall was improved. The precision was reduced from 99% to 76%. This is following the *law of information retrieval.* Concluding, the newly created dataset is more complete, but less relevant. However, overall can be said that more relevant tweets were found.

The classifier did make mistakes. A sample of 200 Tweets from the *not relevant* dataset was taken to see if the Tweets classified as not relevant, were indeed not relevant. When a tweet is incorrectly classified as not relevant, it is called a false negative (fn). On average 2% of the tweets were found to be false negatives.

		pre filter		post filter				
	hashtag assumption	n	acc	n	acc	fn	tweets added	+%
RookieBlue	0.99	29700	0.01	4604	0.71	0.00	2139	+61.61%
US Open	0.98	179015	0.62	167562	0.67	0.02	95992	+134.12%
AustrianGP	0.99	336739	0.52	264944	0.92	0.035	129259	+95.26%
avg.	0.99		0.38		0.76	0.02		+105.39 %

Table 6.7: Results of the final application

6.5 Conclusion and discussion

Finding additional keywords is a good way of for finding new tweets. However, there are some problems that are linked to the use of additional keywords. Some keywords are more specific and delivers more relevant Tweets. There are keywords which only returns relevant Tweets. But, also keywords that return only a few relevant keywords. In this research tweets with an accuracy less than 80% are discarded. This threshold might be a safe decision, however there are cases that a user want to receive all relevant Tweets. In that case even keywords with a low precision are interesting. Of course adding those keywords will increase noise. The classifier might not be able to filter out the 80% irrelevant tweets. In this research the top 10 of keywords are used as additional keywords. Using less keywords might improve the whole dataset, using a bigger net to catch Tweets does not always increase the dataset.

The final application only looks at Tweets where at least one keyword is in the Tweet text. In the datasets are also Tweets which do not contain any of the keywords. In the Game of Thrones dataset 95% of the Tweets did not contain any keywords. For Arrow it was 76%; FrenchGP 15% and for Giro d'Italia it was 12%. Looking at the tweets and the documentation of Twitter [11] some causes were found:

1. Sometimes the keyword can be found in the user description. That can be problematic, because a user can be interested in the hashtag, but it does not ensure that all Tweets of the user are about the topic.
2. Twitter also looks at the link that is shared. When a keyword is found in the title of a webpage, the Tweet will be included as well.
3. When an username is the same as the keyword all posts by that user are returned.

Currently, this research is only looking at keywords in a Tweet text. Other properties, such as username or user description, can improve the classifier. Increasing the amount of features can sometimes solve the sparseness problem of Tweets.

A similar study is executed by Ward [47]. He did his research on TV-shows and found on a accuracy of the whole expanded dataset of 75%. Using automatic query expansion more tweets were retrieved. For each TV-show around 55% more relevant Tweets were found. This research similar results are found.

Concluding, our final application is able to create a more complete and relevant dataset. The choice of keywords is very important, because it can add a lot of noise when chosen wrongly. Fortunately, after filtering the accuracy was usable.

7
Conclusion

Our goal was: *to create a complete and relevant dataset on a topic.* This was executed by identifying additional keywords to retrieve more tweets and by creating a classifier Tweets were filtered in order to create a relevant dataset. In the end the amount of Tweets was increased with 105% with an the accuracy was 76%. The accuracy of using only the hashtag is around 95%. The recall has improved with 105% and the precision decreased with only 19%.

Q1: How to create a more complete dataset on a topic by identifying additional keywords? Additional keywords were defined using three different methods. *TermFreq* did a word count and did perform the worst. Followed by *ProbFact* using the factor between probabilities of a term of relevant Tweets compared against normal Tweets. The last method is called *Overlap* and uses a density estimation of the Tweet volume on Twitter. Using *Overlap* 90% of the top 10 keywords were useful. This was 80% for *ProbFact* and 50% for *TermFreq*.

Q2: How to create a dataset of relevant Tweets on a topic by filtering the dataset? A classifier is needed for filtering a dataset. A classifier is able to automatically classify a Tweet as relevant or not. Two classifiers were selected for the task of filtering Tweets. A Naïve Bayes classifier and a Support Vector Machine. Both classifiers did perform really well on Reuters-21568. The classifiers were also evaluated on Tweets. For Tweets the Bayes Classifiers outperforms SVM. Bayes was selected and had an average precision of 85%.

Q3: How to combine filtering and identifying additional keywords in such a way that it will result in a more complete and relevant dataset? The best performing methods of research question 1 and 2 were combined into a final application and tested on real life events. The method *Overlap* was combined with a Bayes classifier into a program. This program can be bootstrapped by a single hashtag and it will automatically select additional keywords and the receiving Tweets are classified using the Naïve Bayes classifier. Some additional keywords are producing more noise than relevant Tweets. The Tweet containing these keywords are filtered afterwards with a filter.

7.1 Discussion

Other research shows similar results. Ward [47] and Dan & Park [20] are filtering Tweets for TV-shows. Ward is using query expansion to receive more Tweets and with a classifier they had an accuracy of 77%, which is almost the same as this research. Dan & Park are doing a little bit better with an precision of 80%.

Hashtag Using keywords and hashtags can cause some problems. First of all, a hashtag is temporal. A hashtag can refer to multiple events. This is in general not an issue, because most times the event is on a different time. It is advisable to select a hashtag that is directly linked to the original event. An issue by finding additional keywords is that query drifting can occur. For example, looking at the police TV-show *Rookie Blue*. Using the Twitter search some additional keywords were found. A keyword was *rookie*. This words are too general and more than 50% of the dataset was about rookie mistakes or rookie moves. The topic was shifted from a police show of rookie police officers to rookie mistakes.

Error chain In every phase in the final program some error was added. Selecting the best additional keywords adds some noise, because some keywords produce less relevant Tweets. Training the classifier is executed under the hashtag assumption (see section 5.4.1). This assumption is on average 95% correct, also adding noise. The classifier is trained on the hashtag assumption and the classifier itself makes errors. This chained list of errors reduces the accuracy in the whole application. Fortunately, limiting the amount of additional keywords will reduce the error for the keywords finder and using a very specific hashtag will reduce the error on the hashtag assumption. Also a better training set can improve the classifier and even another classifier might improve the classifier.

Scalability A lot of tweets are produced every second. Handling huge amount of Tweets requires architecture that can handle those tweets. This is solved using a *message queue* infrastructure. Multiple writers and/or classifiers can be linked to the streaming API queue in order to handle the huge input stream of Tweets. In this research the Twitter streaming API was the input source, but other sources can be used of course.

Applicability This research focus was Tweets of events and/or campaigns. Such as, the Movember campaign or a TV-show. This could be extended to chatter about another topic. It might also be applicable to a stream of chat messages of a video. Only showing chats that are relevant to the video is one example. Other sources of information can be used as input for the final method. Tweets are limited into 140 characters, which are short messages compared to news articles. The proposed method could also be applied to a stream of news articles or a stream of Facebook messages.

Overall can be said that the final program does return more tweets with a 76% accuracy and a increase of relevant Tweets with 105%. The tool can be found at `https://github.com/haneev/TweetRetriever`.

7.2 Future work

This research looked at finding additional keywords and filtering Tweets. There are some suggestions that can improve our final result, but was not included in this project due to time. We propose several additions that may retrieve more relevant tweets.

Adaptive Twitter is a live and booming platform. A lot of new keywords and hashtag are created everyday. The above described method bootstraps the application once with additional keywords, but newly created keywords are missed, due to the fact that new keywords are invented during the campaign or event. An example was used by Plachouras et al. [40] they show a drift of keyword usage, due to the Boston bombings. The usage of keywords was changed by the users of twitter as a reaction on the bombing. To handle this sudden change, each correctly matched tweet should be added to the keywords method and each 20 minutes the additional keywords method should evaluated, to find new keywords that may have become important. This will create an adaptive method for receiving all related data to certain events or campaign. An addition is to discard keywords that were found to deliver to much noise. This can be measured using classification mistakes and the ratio between correctly classified tweets and incorrectly classified tweets.

An disadvantage of this adaptive streaming method, is that query drifting [14] can occur, because the new event can relate to the original topic, however it can be a topic on itself as well. For example, the Boston bombings are related to the Boston marathon, but when you look only at the runners the bombing might be noise. To prevent sudden change of keywords some thresholds can be set. Using thresholds, the start of the newly hashtag are missed. However this ensures that only important new keywords are taken into account. The main advantage is that this method will adapt to the new situation and will provide more relevant data.

Classifiers Currently, the final application is using Multinomial Naïve Bayes as classifier, because it was easy to train, fast to apply and it can handle many different features. However, Bayes can have at best an accuracy of 90%. Other classifiers might perform better in this situation. The C4.5 classifier used by Ward [47] can perform really well on a few features. As shown in the example of USOpen in section 6.4.2 the training data is very important. When in the training data errors exists this ends up in all the phases of the program and eventually in the dataset itself. This is (again) a balance between usability of simply typing a hashtag or carefully create a proper training set. An addition might be to count the number of occurrences between several hashtag and the co-occurrence of the terms. When a hashtag has a high occurrence, but a very low co-occurrence it might be about a different topic.

Hashtag It is also important to check if a tweet hashtag is really about the topic. Some measure could be created to detect the selectivity/specificity of a tweet. A hashtag can really broad, for example #world. Or narrow: #thebigbangtheory. However, there are some cases in the middle, such as #woods. #woods is in combination with #USOpen about Tiger Woods the golfer, but when the Tweets originated from Green Peace it was about trees. Using the context of the hashtag better keyword selection might improve the additional keywords. Another remark is that a hashtag is time sensitive. Some hashtags in November are about the campaign #Movember, but in March it is about something else. Some information about the time could improve the keyword selection as well.

Goal related To make this research accessible, a nice interface needs to be created in order to command and control the final program. This is needed, because a complete and relevant dataset of tweets can be used by people of many disciplines. Monitoring the stream during the event is also important to adapt and improve the keywords in order to constantly improve the data. For example, the error rate and the success rate per keyword is information that can be used to select keywords.

Bibliography

[1] Reuters-21578 text categorization collection data set. https://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection, 1987.

[2] Interesting fact: more tweets posted are 28 characters than any other length [updated]. http://thenextweb.com/twitter/2012/01/07/interesting-fact-most-tweets-posted-are-approximately-30-characters-long/, Jan. 2012.

[3] Twitter #datagrants selections. https://blog.twitter.com/2014/twitter-datagrants-selections, Apr. 2014.

[4] About twitter. https://about.twitter.com/company, Mar. 2015.

[5] Additive smoothing. https://en.wikipedia.org/wiki/Additive_smoothing, June 2015.

[6] Introducing json. http://www.json.org/, Apr. 2015.

[7] Machine learning. https://en.wikipedia.org/wiki/Machine_learning, Aug. 2015.

[8] Message queue. https://en.wikipedia.org/wiki/Message_queue, June 2015.

[9] Oauth 2. http://oauth.net/2/, Apr. 2015.

[10] The porter stemming algorithm. http://snowball.tartarus.org/algorithms/porter/stemmer.html, June 2015.

[11] Streaming api request parameters. https://dev.twitter.com/streaming/overview/request-parameters#track, June 2015.

[12] H. Abdelhaq, C. Sengstock, and M. Gertz. EvenTweet: Online Localized Event Detection from Twitter. *Proc. VLDB Endow.*, 6:1326–1329, 2013.

[13] H. Achrekar, A. Gandhe, R. Lazarus, S. H. Yu, and B. Liu. Predicting flu trends using twitter data. *2011 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2011*, pages 702–707, 2011.

[14] J. Allan. Incremental Relevance Feedback for Information Filtering. *Baseline*, (August), 1996.

[15] B. Baharudin, L. H. Lee, and K. Khan. A Review of Machine Learning Algorithms for Text-Documents Classification. *Journal of Advances in Information Technology*, 1(1):4–20, 2010.

[16] V. Bobicev and M. Sokolova. An Effective and Robust Method for Short Text Classification. *AAAI*, pages 1444–1445, 2008.

[17] C. Carpineto and G. Romano. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Computing Surveys*, 44(1):1–50, 2012.

[18] C.-C. Chang and C.-J. Lin. Libsvm. *ACM Transactions on Intelligent Systems and*

Technology, 2(3):1–27, 2011.

[19] W. B. Croft, S. Cronen-Townsend, and V. Lavrenko. Relevance feedback and personalization: A language modeling perspective. *In Proceedings of Second DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[20] O. Dan and F. Park. Filtering Microblogging Messages for Social TV. pages 197–200, 2011.

[21] F. Debole and F. Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56(6):584–596, 2005.

[22] C. C. Eugene Agichtein Debora Donato, Aristides Gionis, Gilad Mishne. Finding High-quality Content in Social Media. *International Conference on Web Search and Data Mining*, pages 183–194, 2008.

[23] D. Gayo-Avello. " i wanted to predict elections with twitter and all i got was this lousy paper"–a balanced survey on election prediction using twitter data. *arXiv preprint arXiv:1204.6441*, 2012.

[24] M. Hall, H. National, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software : An Update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[25] M. Hearst, S. Dumais, E. Osman, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28, 1998.

[26] A. Kibriya, E. Frank, B. Pfahringer, and G. Holmes. Multinomial naive bayes for text categorization revisited. *In AI 2004: Advances in Artificial Intelligence*, pages 488–499, 2005.

[27] J. H. Kietzmann, K. Hermkens, I. P. McCarthy, and B. S. Silvestre. Social media? Get serious! Understanding the functional building blocks of social media. *Business Horizons*, 54(3):241–251, 2011.

[28] S. Kinsella, V. Murdock, and N. O'Hare. "I'M Eating a Sandwich in Glasgow": Modeling Locations with Tweets. *Proceedings of the 3rd international workshop on Search and mining user-generated contents - SMUC '11*, page 61, 2011.

[29] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas. Machine learning: A review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190, 2006.

[30] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a Social Network or a News Media ? Categories and Subject Descriptors. *Www 2010*, pages 591–600, 2010.

[31] V. Lavrenko, J. Allan, E. DeGuzman, D. LaFlamme, V. Pollard, and S. Thomas. Relevance models for topic detection and tracking. *Proceedings of the ...*, pages 115–121, 2002.

[32] D. Lazer, D. Brewer, N. Christakis, J. Fowler, and G. King. Life in the network: the coming age of computational social. *Science*, 323(5915):721–723, 2009.

[33] W. Li, P. Serdyukov, A. P. de Vries, C. Eickhoff, and M. Larson. The where in the tweet. *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, page 2473, 2011.

[34] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text Classification using String Kernels. 2:419–444, 2002.

[35] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[36] P. T. Metaxas, E. Mustafaraj, and D. Gayo-Avello. How (not) to predict elections. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference*

on Social Computing (SocialCom), 2011 IEEE Third International Conference on, pages 165–171. IEEE, 2011.

[37] G. Miller. Social scientists wade into the tweet stream. *Science*, 333(6051):1814–1815, 2011.

[38] F. Morstatter, J. Pfeffer, H. Liu, and K. Carley. Is the Sample Good Enough ? Comparing Data from Twitter s Streaming API with Twitter s Firehose. *Association fo the Advanced of Artificial Intelligence*, 2013.

[39] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. *Proceeding of the 17th international conference on World Wide Web - WWW '08*, pages 91–100, 2008.

[40] V. Plachouras, Y. Stavrakas, and A. Andreou. Assessing the coverage of data collection campaigns on Twitter: A case study. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8186 LNCS(Ict 270239):598–607, 2013.

[41] P. K. Prasetyo, D. Lo, P. Achananuparp, Y. Tian, and E. P. Lim. Automatic classification of software related microblogs. In *IEEE International Conference on Software Maintenance, ICSM*, pages 596–599, 2012.

[42] L. Richardson and S. Ruby. *RESTful Web Services.* 2007.

[43] E. Sang and J. Bos. Predicting the 2011 dutch senate election results with twitter. *Proceedings of the Workshop on Semantic Analysis in ...*, (53):53–60, 2012.

[44] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas. Short text classification in twitter to improve information filtering. *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10*, page 841, 2010.

[45] J. Vandewalle. Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[46] M. Wang, L. Lin, and F. Wang. Improving Short Text Classification through Better Feature Space Selection. *2013 Ninth International Conference on Computational Intelligence and Security*, pages 120–124, 2013.

[47] E. Ward. Tweet Collect: short text message collection using automatic query expansion and classification. 2013.

[48] E. Ward, K. Ikeda, Maike Erdmann, M. Nakazawa, G. Hattori, and C. Ono. Automatic query expansion and classification for television related tweet collection. , DBS-155(10):1–8, 2012.

[49] J. Xu and W. B. Croft. Query expansion using local and global document analysis. *SIGIR '96: Proceedings of ACM SIGIR Conference*, 19:4, 1996.

[50] S. Zelikovitz and S. Zelikovitz. Improving Short-Text Classi cation Using Unlabeled Background Knowledge to Assess Document Similarity. *Test*, 1999.

[51] H. Zhao and Q. Zeng. Micro-blog keyword extraction method based on graph model and semantic space. *Journal of Multimedia*, 8(5):611–617, 2013.

[52] W. X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E.-P. Lim, and X. Li. Topical keyphrase extraction from Twitter. *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 379–388, 2011.

Appendix

A. Raw results of Overlap tool validation

In the table 1 the raw numbers are shown. The combination of data for 2013/2014 for keyword 1 and 2 are known and compared to the estimating overlap function. The results of the tool are validated against the Movember data collection in 2013/2014. The result of each query (q) are compared with the real results. The movember dataset contains of tweets containing the word movember. The amount of tweets containing the word *movember* is 1.6 million. We will count the amount of tweets containing *movember and the word w*. We will compare this to the method by submitting a boolean query at Twitter. The query will be *movember AND w*.

The error is expressed in the std deviation shown below. On average the tool guesses about right, but the std dev is around 31%.

Keyword 1	Keyword 2	Real	Estimate	Error in Tweets	%	distance to 1
cancer	mamming	1045	1843	798	0.5670	0.4330
cancer	movember	33062	33314	252	0.9924	0.0076
cancer	shaven	229	714	485	0.3207	0.6793
cancer	breast	3125084	2080916	1044168	1.5018	0.5018
cancer	sunsmart	353	500	147	0.7060	0.2940
cancer	pinkribbon	2759	4479	1720	0.6160	0.3840
cancer	sunscreen	16825	12435	4390	1.3530	0.3530
cancer	tash	897	1811	914	0.4953	0.5047
cancer	moustache	5429	5629	200	0.9645	0.0355
cancer	shave	41229	27669	13560	1.4901	0.4901
cancer	ironman	2682	3066	384	0.8748	0.1252
cancer	awareness	1460618	890830	569788	1.6396	0.6396
cancer	breastcancer	78873	2073938	1995065	0.0380	0.9620
cancer	november	33106	25659	7447	1.2902	0.2902
cancer	donation	61167	48970	12197	1.2491	0.2491
cancer	spf	2791	2603	188	1.0722	0.0722
movember	noshave	896	1147	251	0.7812	0.2188
movember	raising	10625	8322	2303	1.2767	0.2767
movember	december	6321	5985	336	1.0561	0.0561
movember	shaved	7978	10106	2128	0.7894	0.2106
movember	prostate	17471	12650	4821	1.3811	0.3811
movember	sporting	4158	3704	454	1.1226	0.1226
movember	moustache	97504	68278	29226	1.4280	0.4280
movember	itchy	2093	1969	124	1.0630	0.0630
movember	nicely	3707	3969	262	0.9340	0.0660
movember	menshealth	17861	18025	164	0.9909	0.0091
movember	shaven	4543	5181	638	0.8769	0.1231
movember	shave	42115	32174	9941	1.3090	0.3090
movember	bearded	1290	1255	35	1.0279	0.0279
movember	mo	70288	163069	92781	0.4310	0.5690
movember	awareness	27319	22385	4934	1.2204	0.2204
movember	facial	19980	19695	285	1.0145	0.0145
movember	mustaches	12780	9982	2798	1.2803	0.2803
movember	mustache	53021	66259	13238	0.8002	0.1998
movember	donation	10788	15544	4756	0.6940	0.3060
movember	mobros	4236	8834	4598	0.4795	0.5205
movember	october	7353	2675	4678	2.7488	1.7488
movember	tash	15536	12879	2657	1.2063	0.2063
movember	donating	13426	54156	40730	0.2479	0.7521
movember	fuzz	1199	1312	113	0.9139	0.0861
movember	snor	5671	7262	1591	0.7809	0.2191
movember	goodcause	1242	989	253	1.2558	0.2558
movember	enlisted	25079	26740	1661	0.9379	0.0621
movember	beard	30294	29078	1216	1.0418	0.0418
Average				1.005	0.3135	
Std. dev				0.3135	0.2111	

Table 1: Raw results overlap tool

List of Tables

List of Figures