Twitter search: Building a useful search engine.

Hurlock, Jonathan

Twitter Search: Building a Useful Search Engine

Jonathan Hurlock

Submitted to Swansea University in fulfillment

of the requirements of the Degree of Doctor of Philosophy

Prifysgol Abertawe
Swansea University

Department of Computer Science

Swansea University

2015

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed . ... (candidate)

Date: 20th November 2015

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended

Signed ..... (candidate)

Date: 20th November 2015

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organizations.

Signed ... (candidate)

Date: 20th November 2015

Abstract

Millions of digital communications are posted over social media every day. Whilst some state that a large proportion of these posts are considered to be babble, we know that some of these posts actually contain useful information. In this thesis we specifically look at how we can identify reasons as to what makes some of these communications useful or not useful to someone searching for information over social media. In particular we look at what makes messages (tweets) from the social network Twitter useful or not useful users performing search over a corpus of tweets.

We identify 16 features that help a tweet be deemed useful, and 17 features as to why a tweet may be deemed not useful to someone performing a search task. From these findings we describe a distributed architecture we have compiled to process large datasets and allow us to perform search over a corpus of tweets.

Utilizing this architecture we are able to index tweets based on our findings and describe a crowdsourcing study we ran to help optimize weightings for these features via learning to rank, which quantifies how important each feature is in understanding what makes tweets useful or not for common search tasks performed over twitter. We release a corpus of tweets for the purpose of evaluating other usefulness systems.

Published Work

Work contained within this thesis has been published in the following:

Peer-reviewed Conference Papers

Hurlock J. and Wilson M. L. 'Twitter Search: Separating the Tweet from the Chaff' In International Conference on Weblogs and Social Media, ICWSM 2011. AAAI

This work is detailed in Chapter 3.

Acknowledgements

I would like to thank the following people for their guidance and support through the process of this Ph.D.

Firstly and foremost, I would like to thank Matt Jones and Max Wilson for their guidance, support and encouragement. Without both of you, I would probably have never gotten to this point, thank you both. I would also like to thank you for all the extra support you gave me, as I know my Ph.D. has not been the smoothest of rides.

I would also like to thank Mat Wilson, thank you for the all the help and entertainment you provided through this long journey.

I would also like to thank all those in the FIT Lab who helped at the final hurdle, especially Emma James, Tim Neate, Jennifer Pearson for reading through my horribly dyslexic chapters and providing advice.

Finally, I would like to thank my parents for supporting me whilst I undertook this task.

This work is part-funded by the European Social Fund (ESF) through the European Union's Convergence programme administered by the Welsh Government

Table of Contents

Chapter 1 : Introduction

Everyday millions of posts are made to social media services and this number seems to be rising (Twitter, Inc., 2011). Finding useful information in this vast expanse of data is a hard task; this sentiment has been echoed by creators of services such as twitter.

> "I think the challenge not only for Twitter , but for the technology industry at large. Is building more relevant filters, in real time. Like *being able to surface valuable information immediately, No matter who it is, who's listening or who's broadcasting, it is a really really hard problem,* and it makes Twitter a lot more meaningful[...] We've gotten really really good at being able to put content in, into media[...] getting it out in a relevant, valuable way, in real time is still very difficult."
>
> - Jack Dorsey (Creator of Twitter) (Dorsey, 2011)

There are various stakeholders who are interested in getting meaningful data out of social media. Twitter has been used in all types of scenarios across a range of areas. We know for instance Twitter has been used as a question and answering service (Morris, Teevan, & Panovich, 2010). It has been used to reshape healthcare (Hawn, 2009). Researchers and healthcare professionals are using twitter to analyze public health and to track epidemics (Aramaki, Maskawa, & Morita, 2011) (Lampos, De Bie, & Cristianini, 2010) (Culotta, 2010). We have seen how twitter has been used in disaster events, (Qu, Huang, Zhang, & Zhang, 2011) (Vieweg, Hughes, Starbird, & Palen, 2010) delivering news (Kwak, Lee, Park, & Moon, 2010), and offering advice and current updates (Imran, Elbassuoni, Castillo, Diaz, & Meier, 2013) (British Broadcasting Corporation, 2013). Researchers have also used twitter to correlate mood towards companies and stock prices (Boolen, Mao, & Zeng, 2011). We have seen how twitter has been utilized to identify new news stories, as well as tracking online events. (Petrovic, Osborne, & Lavrenko, Streaming first story detection with application to twitter, 2010) (Phuvipadawat & Murata, 2010) One of the most obvious

uses of twitter is in the marketing domain, for interacting and analyzing perceptions to brands, products and services.

Hurlock and Wilson (Hurlock & Wilson, Searching Twitter: Separating the Tweet from the Chaff, 2011) conducted an experiment that asked users to rate Twitter search results pages with scores out of 5 in terms of relevancy. Overall the mean score for all rated tweets over three common search tasks performed on microblogging data sets was 2.2. Indicating very low relevancy scores, and in one type of task it was as low as 1.25. The work carried out by Hurlock and Wilson was a main motivation behind this work.

Whilst papers such as Earlybird: Real-Time Search at Twitter (Busch, Gade, Larson, Lok, Luckenbill, & Lin, 2012) and Evaluating real-Time Search over Tweets (Soboroff, McCullough, Lin, MacDonald, Ounis, & McCreadie, 2012) have concentrate on the speed at which the system can ingest content rapidly and make it searchable immediately, very few have looked at how users of such a service and data type judge the information that is retrieved.

McCreadie and MacDonalad (McCreadie & MacDonald, 2013) acknowledge that finding tweets can be a challenging task. However, they state that the relevance of a tweet is dependent both on its content and whether it links to a useful document, without any reference to research being carried out on this factor.

Work by Naveed et al. (Naveed, Gottron, Kunegis, & Alhadi, 2011) suggests that retweets reflect what the Twitter community considers interesting on a global scale, and suggests that it can be used as a function of interestingness. In their work they also suggest features that contribute to the likelihood of a retweet, and thus identifying interesting tweets, they weight these features via learning to rank utilizing logistic regression and in their future work intend to use interestingness as a static quality measure for IR on microblogs.

Work by Cherichi et al. (Cherichi & Faiz, 2013) and Jabeur et al. (Jabeur, Tamine, & Boughanem, 2012) have explored microblog IR by splitting tweets into different

features such as content relevant features, tweet relevance features and author relevant features, however there is no explanation of where these features were derived from.

The TREC microblog track assumes that the best result set for a given query is to "return the most recent but relevant information" (Ounis, MacDonald, Lin, & Soboroff, 2011).We know that temporal information is a big part of microblog search by classifying tweets via search logs (Teevan, Ramage, & Ringel Morris). However we also know there are other types of search tasks performed on Twitter data.

In this Thesis we wish to investigate what really makes tweets useful or relevant to users performing microblog search. Is the most recent tweet really the most important factor to a user or are there other more important factors? Manning et al. (Manning, Raghavan, & Schütze, 2008) state the following:

> "...in the final analysis, the success of an IR system depends on how good it is at satisfying the needs of these idiosyncratic humans, one information need at a time."
>
> - Manning et al. (Manning, Raghavan, & Schütze, 2008)

Whilst classically IR takes the view that users are searching to fulfill an information need and that information need (Schneiderman, Byrd, & Croft, 1997) is either fulfilled successfully or not (binary judgment).

We now know that depending on the type of search users are performing, searches are not just to satisfy an information need, but users also have other intents when performing search.

In web search we see such intents that include transactional and navigational search. (Broder, 2002) We now know the intent behind a web search query may not at all be informational in nature. Andrei Broder found via query log analysis that queries could be classified into Navigational (20%), Informational (48%) and Transactional (30%).

Elsweiler et al. (Elsweiler, Wilson, & Kirkegaard Lunn, Understanding casual-leisure information behaviour., 2011) have looked at a newly identified type of search 'casual-leisure' search; exploratory search scenarios where by the goal is not information-oriented. Although not classically seen as IR Morris et al. (Morris, Teevan, & Panovich, 2010) have looked at the types of questions that were asked on social networks, and whilst most of informational in context, queries were asked through social networks regarding recommendations, opinions, factual knowledge, rhetorical questions, invitations, favors, social connections and offers

Jansen et al. (Jansen, Spink, & Saracevic, 2000) argue that internet search is very different from IR searching as traditionally practiced and researched. The reasons behind searching and the way people are searching has evolved. Not only have intents changed, but also the way in which people perform search, tools such as keywords clouds, tag clouds and faceted search have enabled people to move away from traditional keyword search.

We wish to explore how users perceive what makes a result useful to them when performing one of the three most common search tasks on Twitter.

This Ph.D. has been partially funded by an industrial partner. Originally called Kaimai Research based in Swansea, this company was acquired by Adzeee during the duration of this Ph.D. Adzeee offers a dynamic publishing platform enabling content monetization via the sponsorship of an intelligent information retrieval service, which produces actionable 'zero latency' knowledge. As such the motivation of this project was to create a search engine that could be utilized by Adzeee to identify useful tweets to users. Allowing them to target users who provide the best information, or are mostly to be involved with further interactions with a client. The relationship with author and Adzeee is via a Knowledge Economy Skills Scholarship (KESS). KESS is a major European convergence programme led by Bangor University on behalf of the HE sector in Wales. Benefitting from European Social Funds (ESF). KESS supports collaborative research projects (Research Masters and PhDs) with external partners based in the Convergence area of Wales. As part of the KESS scheme students must acquire credits via attending and presenting at conferences, attending a KESS Grad

School, submitting monthly and quarterly reports based on progress as well as spending a 3 month secondment inside of the host company.

As a result of this collaboration the project attempts to help identify useful tweets, by introducing a novel set of filtering features for both useful and non-useful tweets, as well as describing an architecture on which this detection can run.

Research Contributions

Through evaluation of existing literature, and systems in the wild we have identified a need to

- Understand what factors make a tweet useful to people performing searches of microblogging data
- Develop a robust framework for indexing and retrieving large amounts of microblogging data in a timely fashion
- Create a test corpus to allow us to compare our system against others
- Produce a system whereby we can index large datasets, and retrieve data in a timely fashion, and automatically whether a tweet is to be deemed useful or not for a given query

Chapter Outline

In this section we give a brief overview of the thesis and what you reader can expect to read in each chapter.

Chapter 2: This consists of a literature review as well as background information that provides allows the reader to understand the rest of the thesis. We provide an overview of Twitter and information retrieval. We also give an overview of research performed on Twitter.

Chapter 3: Describes a study we conducted that aimed to find what factors made tweets useful or not useful to people performing searches over a microblogging data. We based this upon the three most common types of search performed over microblogging platforms.

Chapter 4: Describes both the physical and software architecture we chose that has allowed us to build a scalable information retrieval system for operating over large data sets. We describe components and methods we evaluated as well as rationale as to why we chose certain components. This section acts as a guide for anyone else wishing to build a similar system and wishes to know the benefits of certain components, as well as how components interact with each other.

Chapter 5: In this chapter we describe how we programmatically extract features/codes from tweets that we found in Chapter 3, and how they interact with the system described in Chapter 4. These features/codes will then be utilized in Chapter 7. To help optimally rank tweets in terms of usefulness.

Chapter 6: This chapter describes a corpus of tweets we have created via crowdsourcing for the purpose of evaluating retrieval systems looking at usefulness. Whilst there is an interest in performing IR over twitter data, there is no dataset that provides researchers with both tweet ids, as well as user judgments scores for usefulness. We have built the first dataset that has both of these metrics available for users to download and utilize. This is helpful for other researchers working on this or similar tasks allow them to compare their system to ours.

Chapter 7: In this chapter we utilize the work carried out in previous chapters to automatically assign weightings to the features we extracted in chapter 5. The main contribution of this chapter is to inform the reader as to which factors are most important and importantly how important are they are, to making a tweet useful or not useful.

Chapter 8: Concludes the thesis by summarizing the contributions of the thesis as well as giving possible avenues for future work.

Chapter 2: Literature Review

In the previous chapter, we identified the need for a search service to provide support for users wishing to identify useful information in a microblogging environment.

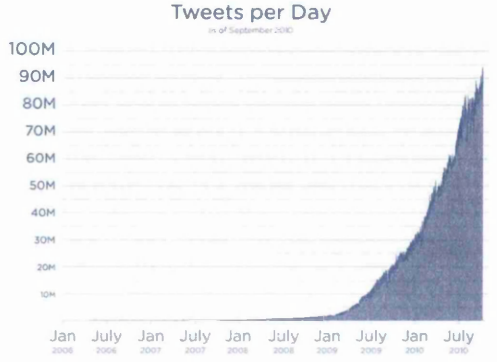
In this section we provide the reader with an overview of the research landscape concerning the problem we wish to address. We touch on several fields in computer science, and how they have gone about trying to tackle similar or relevant problems. As well as an overview of the research landscape we briefly give an overview of Twitter, and some of the conventions used, so the reader is aware of some of the terminology we use throughout this thesis.

2.1 A Brief Overview of Twitter.

Twitter is a microblogging service that allows users to post 140 character posts onto their profile. It is set at 140 characters for historical reasons relating to the length of a SMS (Short Message Service) text. Users can follow other user's posts, though this network does not necessarily require 'mutual following' unlike other social networks.

2.1.1 The Shape of Twitter Data

Twitter started in March 2006, with Jack Dorsey sending the tweet, "just setting up my twttr" (Twitter Inc., 2014), since these humble beginning Twitter has seen explosive growth, with the last confirmed reports saying 400 million plus tweets were being sent per day (Twitter, Inc., 2013), this growth can be seen in the graphs below.



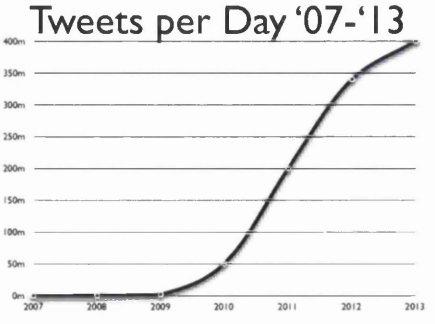


Figure 2.1 *Shows Tweets per day from January 2007 to 2010 on the left hand side* (Twitter, Inc., 2010)*Whilst the graph on the right hand side shows estimated tweets per day based off of press releases via the Twitter blog*[1].

When we think of a tweet, we think of a 140 character message, with some associated data about the user. However, as discussed there is a lot more data "underneath the bonnet", that most users don't see.

In terms of data, it was reported Twitter users were generating 12 terabytes of new data per day in September 2010.[2] However, since this report we know that Twitter is seeing over 4 times as many tweets being sent per day as of late 2013 (as seen in Figure 2.1). This introduces one of one of our biggest problems, how do we deal with this much data and growth?

2.1.2 The Language of Twitter

However, like many services the use of Twitter has evolved and as this transformation has taken place certain service specific traits have evolved. We describe some of these Twitter specific language traits in this section.

2.1.2.1 @Mentions/@Replies

@Mentions are a way of identifying another user in a piece of text, for instance the following tweet identifies that the user @FITLab_Swansea, is referencing the user @dodopat

Figure 2.2 *Tweet demonstrating an @mention.*

[1] http://blogs.twitter.com/

[2] http://www.neowin.net/news/storing-tweets-requires-four-petabytes-of-data-a-year

When a user @mentions another user. The user who is referenced, will receive a notification that they have been referenced in a tweet.

There are however, some cavets to @mentions. If a tweet starts with a @mentions such as that below from the Tesco user account mentioning the user @keewa.

Figure 2.3 *A conversation on Twitter between @keewa and @Tesco. @Tesco have used the @mention construct to reply to @keewa.*

The user @keewa will receive a notification that they have been mentioned by @tesco. However, by default no one else will see this tweet unless they are both a follower of @keewa and @tesco. This is due to it being part of a conversation between those two users.

Other users of Twitter may find these tweets via search, or by going to the @tesco account and selecting to view 'tweets and replies'.

However, this conversation behavior can be countered by putting any character before the '@' symbol. It is common to see a full-stop/period before the @mention as shown in the example below.

Figure 2.4 *A conversation on Twitter between @BrightonDennis and @stephenfry. @Stephenfry has used the '.@mention' construct publically reply to @BrightonDennis.*

2.1.2.2 $Companies

As well mentioning other users, there is a convention oh how to mention companies via Twitter. This is done by prefix the companies NASDAQ code with a dollar sign, for instance if we wanted to mention Twitter Incorporated we would tweet something with $TWTR, TWTR being the NASDAQ code for Twitter.

This is not a widely used convention, and is limited to US based companies based on the NASDAQ exchange.

We have provided an example of this behavior in use in the following tweet, where the financial times have referenced Google via the $GOOG code and Apple via the $AAPL code.

Figure 2.4 *Example of the $Companies tag being used to indicate Google ($GOOG) and Apple Inc. ($AAPL).*

2.1.2.3 #Hashtags

As well mentioning users and companies. Twitter has a convention for tagging a tweet about a topic. This is called hashtagging. A word or Phrase (normally written in CaMeL case) is used to denote a topic, an example of this can be see in Figure 2.3 denoting the hashtag #WeLoveFreshDoughnuts.

These conventions whilst popular on Twitter, are also popular on other social media sites. Facebook and other large social networking sites have started to incorporate this behavior and functionality in their services.

2.1.2.4 Retweets

Retweeting is a behavior found on Twitter. Which is where a user creates a post and if another user wishes to 'share' their post on their wall, they retweet the original post. (Twitter Inc.)

There are two common ways this is achieved. The first is by using the retweet button presented on all tweets. This can be seen in figure 2.4, it is the icon which is located next to the star on the bottom right. This then posts this selected tweet, to the user's page unedited, and show the author as the originating author.

Another convention is by using prefixing a tweet with the characters RT followed by an @mention to the original author, then followed by the message. This is normally done to then add a comment to the originating tweet.

2.1.3 Twitter Metadata

As well as in messages, each tweet contains metadata. Whilst a lot of this is to do with the person who has sent the tweet, it also contains data about, where and when tweets were sent, depending on the user's privacy options.

If the tweet is part of a conversation or a direct message, it contains information about the intended recipient, and if the tweet contains rich media via a Twitter accredited media provider, it can also include data about the media to be bundled with the tweet.

Whilst most users of the service will only see the 'surface' of a tweet, there is a surprising amount of data that lies underneath. Raffi Krikorian's white paper (Krikorian, 2010) explains a lot of the early metadata included in tweets, however, as Twitter has grown this structure has changed, and object have been added, removed or modified. (Twitter Inc.) Based on our work in Chapter 6, tweets collected ranged between 2-9kb in size.

2.2 Relevant Research on Social Media & Microblogs

Whilst the definition of social media varies depending on whom you are talking to, and the purpose of what you are talking about. From a business prospective social media may be synonymous with consumer-generated media, however to others social media might be the interactive dialogue which is enabled through web-based communications.

Social media is huge area of interest to researchers, with conferences and workshops being dedicated to understanding, and facilitating social media.

One of the largest social media services at time of writing is Facebook. With more than 500 million active users (Zuckerberg, 2010), if Facebook was a country it would be the third most populated country in the world, following China and India. An average user on Facebook will create 90 pieces of content per month, with more than 30 billion pieces of content shared each month Facebook (Barnett, 2010), we can see that Facebook can provide an interesting insight into how people interact with each other.

Social media encompasses everything from blogs, microblogs, social networking sites, question and answering sites, review and opinion sites, social news and media sharing sites and apps. Microblogs are a form of blog. However, unlike normal blogs, there are restrictions placed on the content, this restriction is to with the size of the content. A microblog entry may consist of just a few words or an embedded video or an image.

Microblogs force users to try and convey an idea, thought, pieces of information in a very concise way. This leads to computation problems in terms of natural language understanding. One might think that because the content is concise it would be easy to pick out meaning, or perform entity disambiguation, however there may be no surrounding content to help perform entity disambiguation, and due to the language (in terms of misspells/text speak) and multilingual nature of Twitter it can be hard for a machine to understand what a tweet is trying to convey.

Due to their concise nature, microblogs allow people to quickly send up-to-date information to one another or share it to the public domain. Twitter has allowed users to beat the traditional press to printers. Breaking and spreading news via Twitter has almost become a normal practice. With stories such as the US Airways 1549 ditching in the Hudson river (Beaumont C. , New York plane crash: Twitter breaks the news, again), to the Mumbai terror attacks (Beaumont C. , Mumbai attacks: Twitter and Flickr used to break news, 2008), Iranian protests (Grossman, 2009), Egyptian and other Middle East/North African political problems have been shared before traditional media has had a chance to write a 'full' story about the situation (Beaumont P. , 2011).

Much research has gone into how we can detect and use data from these events, we describe some of the research in the following section

2.2.1 Helping People Utilize Twitter Data

Individuals and organizations read and publish tweets. In this chapter we have briefly touched upon examples of how Twitter data had been used by different kinds of people in an array of situations. We know journalists use Twitter for finding and relaying content, organizations use twitter for promoting their products or services, we know that government agencies also use twitter to gather information and communicate with the public.

Research has been conducted on how Twitter data is being utilized to inform decisions and to help users. In this section we give examples of some of the work carried out that has utilized twitter as a means to helping people with certain tasks.

In a study conducted by Zhao and Rosson (Zhao & Rosson, 2009) participants were said to have found Twitter posts more valuable than other media for connected information to personal goals. The main reasoning behind this was due to the near real-time nature of the service. Allowing them to keep a "pulse" on people or events they do not encounter in their daily lives.

Aiding Journalist Inquiry

Diakopoulos et al (Diakopoulos, Naaman, & Kivran-Swaine, 2010) aimed to explore how social media can inform journalistic inquiry surrounding large-scale broadcast news events. They sought to understand to what extent Vox Civitas (Diakopoulos & Shamma, 2010) facilitates the detection of insights, analysis and other activities can be obtained through the support of such as system in relation to journalistic inquiry.

The study presented by Diakopoulos et al. concentrated on the 2010 U.S. State of the Union presidential address. Instead of using manual tagging of tweets to detect sentiment, the authors applied a supervised learning algorithm trained with 1900 manually tagged tweets from the state of the union corpus.

Not only did they use automatic extraction of sentiment, but they also calculated relevance scores, uniqueness scores and keyword extraction.

Relevance was calculated by calculating term-vector similarity of the tweet to the moment in the event during which the message were posted. The authors did this by comparing the tweet's message, to a transcript of the event. They state that transcripts for large-scale news events such as the State of the Union are readily available from news services.

Uniqueness is said to be something which may appear to be "unusual" in a social media stream, the authors make a comparison that something's "newsworthiness" often adopts the importance of the unusual or unexpected nature. The authors created a uniqueness metric, to see how a message compares to that of other messages sent during a similar time period.

Keyword extraction was used to identify keywords in the social media stream that could be useful and interesting for guiding analysts. The system aimed to extract descriptive keywords for each minute of the aggregate message content. This was done by extracting the top ten keywords ranked by TF-IDF for each minute of the speech.

Finally Vox Civitas performed sentiment analysis to inform analysts understanding of the popularity of the social media reaction to an event. A two step procedure was used to classify the sentiment of the tweets. Firstly a simple classifier based on a lexicon of words that classified messages based on a lexicon of words that classified messages based on whether they were carrying subjective (positive or negative) information was run over the tweets. Secondly a supervised learning algorithm, which had 1900 manually tagged messages from the State of the Union corpus was run over the dataset.

The authors found the combined classifier resulted in a 5-fold cross validated accuracy of 62.4%, which they state is sufficient for giving an overall impression of the sentiment, however they do note that the classifier fails on difficult cases, such as those involving sarcasm or slang.

When presenting the results of the sentiment in the user interface, sentiment is only represented as an aggregate, this is due to the authors concerns over the accuracy of the sentiment classifier. Also the authors note that sentiment is not displayed for individual tweets, as the authors assume users can quickly surmise sentiment as they are skimming through the tweets.

Aiding Exploration

Work by Bernstein et al. (Bernstein, Suh, Hong, Chen, Kairam, & Chi, 2010) have explored the visualization of data from Twitter, on top of a exploratory system. Currently most systems present lists of results in a reverse chronological order similar to that of traditional web results. Eddi (the system developed by Bernstein et al.) explored other ways of presenting results and trying to aid exploration, by aiming to

allow users to browse items of interest. Through user evaluation users found it to be more efficient and enjoyable way to browse an overwhelming status update feed than standard chronological interfaces.

Identifying and Recommending Interesting Content

Omar et al. (Alonso, Marshall, & Najork, 2013) attempted to address two questions. The first question was how to develop a reliable strategy that resulted in high-quality labels for collections of tweets, and the second question was concerned with asking whether the authors could use labeled collections to predict a tweet's "interestingness".

The authors believed if human judges could label tweets' interestingness effectively they could produce a training set that distinguishes between interesting and uninteresting tweets. From this belief they then said it would be possible to implement a classifier that would use the predictive features from the training set to identify interesting tweets within a dynamic collection.

Omar et al. noted that judging whether a tweet is interesting or uninteresting is a complex and subjective activity, with many factors at play.

When defining the notion of interestingness, the authors maintained a flexible notion, and explored its many interpretations. We have taken a similar approach in our work as not to define what usefulness is. Allowing users in further chapter to interpret it as they wish.

Unlike the studies we performed in later chapters, Alonso et al, recruited participants who were familiar with Twitter. Participants were also recruited from two different crowdsourcing platforms, one which specialized in relevance judgments and the second were recruited from Amazon's Mechanical Turk (AMT). Alonso et al, saw the AMT workers as a proxy for Twitter users with diverse perspectives.

Alonso et al. saw participants classify several sets of tweets with different pre-existing categories over 3 studies, throughout the studies Alonso et al found low levels of label

agreement. So in the final study they attempted to try and get workers, to articulate why they were assigning certain labels. Also witnessed that workers could quickly assess whether a tweet was interesting to a broad audience. However, workers found it difficult to describe why this was when trying to assess whether a tweet was interesting to a broad audience. When assessing whether a tweet is only interesting to a limited audience, workers had less difficulty.

Alonso et al. state that try to identify interesting tweets is a difficult task, as it varies with the judges' own interest and proclivities. In the end they found that a binary labelling scheme was the most tractable for workers. As well as this, they discovered that the best performance came from a small number of very experienced judges rather than a large number of diverse judges.

Due to the low inter-rater agreement on whether a tweet is interesting or not, Alonso et al, concluded that interestingness is indeed a fully subjective notion, stating "there is little hop in constructing a classifier that identifies such tweets".

In the final stages of the paper, the authors examined the correlation between 13 predictive features and a tweet dataset, labeled by the crowd workers. Alonso et al. found that a link's presence is a strong signal of interestingness. We will discuss the idea of a link being present a factor in a tweet's usefulness or making a tweet not useful in chapter 3. Alonso et al. also found that features such as tweet length (without @ mentions) and average BM25 on Twitter queries are also important indicators of quality. We did not find any codes to with tweet length being an indicator of usefulness, however, we draw comparisons with the BM25 and tweet's being TF-IDF relevant in our own work

Evaluating the Value of Microblogging Content

Andre et al. (Andre, Bernstein, & Luther, 2012) wanted to understand how people assigned value to tweets (worth reading, not work reading, middling), contributing an analysis of microblog content from the reader's point of view.

This draws similar parallels with our work, in that we take a consumer point of view, designing a service which is targeting content which we believe will be useful for a given user. The authors created a website that allowed users to obtain anonymous feedback of their own tweets if they agree to anonymously rate tweets by other Twitter users. In total the site gathered 43,738 ratings for tweet, from 21,014 user accounts.

Of the tweets judged, just 36% were considered to be worth reading by users, whilst 25% were not worthy of reading and finally 39% elicited no strong opinion (neutral). As part of this study, the authors looked at categorically labelling each of the tweets utilizing a crowdsourced approach. Using an adapted version of Naaman's (Naaman, Boase, & Lai, 2010) tweet categorization scheme authors obtained a 0.62 moderate agreement Cohen's kappa score, but said they could obtain a 0.81 inter-rater reliability score if they would be allowed to include multiple categories per tweet..

As part of the study they looked at which categories were considered valuable (Question to Follower, Information Sharing and Self-Promotion) as well as what categories were strong disliked (Presence Maintience, Conversation and Me Now)

Authors suggested that outcomes from this research might help design tools to help filter and display content in the future, as well as providing a feedback to users about their perceived value, audience reactions and emerging norms.

First Story Detection

In the paper Streaming First Story Detection with application to Twitter, Petrovic et al. (Petrovic, Osborne, & Lavrenko, Streaming first story detection with application to twitter, 2010) adapt the locality-sensitive hashing (LSH) algorithm to perform first story detection on such a large and high transactional stream of data.

LSH is a randomized technique to perform a nearest neighbor search in vector space, by reducing the amount of time need to perform the computation compared to other nearest neighbor search algorithms.

Authors found by applying LSH in its original state, the algorithm performed poorly and had a high variance in results. The authors modifications to the algorithm both improved the performance and reduced the variance of results.
As well as performing first story detection, the algorithm also proved useful for detecting 'spam' tweets. The researchers also found the two following insights regarding users and news on Twitter:

The number of users that write about an event is more indicative than the volume of tweets written about it.

News about deaths of famous people spreads the fastest on Twitter.
To see how well their improved LSH algorithm worked, the authors compared it to an existing FSH system, in particular the UMass system. The systems were compared on their performance using detection error tradeoff (DET) as a measure, where both systems proved to be very similar, with UMass scoring 0.69, and the improved LSH scoring 0.70, the systems were also compared in terms of processing time, in this test the improved LSH algorithm showed to prove its power, whilst the UMass showed an almost linear time to process documents, the improved LSH algorithm showed a near constant processing well below that of the UMass system.

Whilst this research was very algorithmically heavy, we can see how it would be applied in aiding journalistic inquiry, but also in warning systems.

2.2.2 Why and How People Search Twitter

In this section we introduce the reader to some of the research that has been carried out regarding search in a microblogging environment. The following quote is very relevant to the state of research surrounding microblogs.

> "Research into microblogger's motivations, habits and strategies is in its infancy and our understanding of people's information behavior with respect to microblogs remains murky"
>
> - (Efron & Winget, 2010)

Work by Elsweiler and Harvey (Elsweiler & Harvey, Engaging and maintaining a sense of being informed: Understanding the tasks motivating twitter search, 2014) has tried to tackle the above, by reveal numerous characteristics of Twitter search that differentiate it from more commonly studied search domains, such as web search. Elweiler and Harvey found difficulties encounter by users as well as trying to understanding how and why people search for content. Work by Teevan et al such as #TwitterSearch: A Comparison of Microblog Search and Web Search by Teevan et al. (Teevan, Ramage, & Ringel Morris) and Questions are Content: A Taxonomy of Questions in a Microblogging Environment by Efron & Winget (Efron & Winget, 2010) also look at how and why people perform search over microblogs.

#TwitterSearch: A Comparison of Microblog Search and Web Search explores the differences that occur between web search and microblog search. The paper explores the search behavior of users via the analysis of large-scale query logs, and supplemental qualitative data to explore search behavior on the popular microblogging site Twitter, and the Bing search engine.

The authors identified that information seekers used Twitter to find temporally relevant information such as breaking news, real-time events, and trending topics. As well as this, Twitter was used to find information related to people, examples given are that of content directed at the searcher, information about people of interest (celebrities) and general sentiment and opinion.

The authors compared structural concepts and behavioral qualities of the information seeking behavior of users, Examples given are that of query length the user inputted into the search dialogue box, as well as how users repeated queries to monitor the associated search results.

Findings in the paper indicate that queries targeted at searching Twitter, are less likely to evolve as part of a user session unlike web search. The findings of the study also show how users are using specialized syntax and operators in their search queries when searching Twitter, with 24.23% of Twitters queries either containing an '@' or '#' symbol, whilst prior large-scale log analysis of web search found that only 1.12%

of web queries contained advanced search operators or syntax such as '+','-', quotations or 'site:'.

As well looking at queries the paper explored the results that were shown to users. One interesting result was that 34% of all the Twitter results returned contained an external link. Analysis was conducted to compare the similarities between tweets returned and web snippets returned. Machine learning techniques were used (specifically latent dirichlet allocation - an unsupervised latent variable topic model) to calculate the similarities between words. Twitter topics were found to include more social chatter and current events, whilst web topics tended to contain more basic facts and navigational results.

The authors identified that measures based on term overlap such as TF-IDF tend to be noisy because of the results short length. Although the paper does not describe information retrieval algorithms it does present valuable information regarding the search behavior of people performing searches on Twitter. It also highlights what people are using Twitter for, with regards to search tasks.

2.2.3 Using Twitter as a Question and Answering Forum

As well querying twitter through traditional web forms, Twitter like many other social networks is used as a question and answering forum. Question asking in microblogging environments and online in general is a large and very active research topic. Twitter has even acted as a de facto social search system according to Evans and Chi. (Evans & Chi, 2008)

Efron and Winget (Efron & Winget, 2010) have analyzed characteristics and strategies that people presented when asking questions in a microblogging environment. As part of this analysis, the authors were able to propose a taxonomy of questions asked on microblogs. They were also able to look at why users asked questions on microblogs and what kind of information task they were trying to complete.

A very interesting and revealing finding that came from this research was that, question asking in microblogs is strongly tied to peoples' naturalistic interactions and that the act of asking questions in Twitter is not analogous to information seeking in more traditional information retrieval environments.

The authors created two corpora, a general corpus consistsing of 2,022,544 tweets which were collected via Twitters streaming API, and a community corpus which consisted of tweets written mainly by people who are interested in issues related to information and their friends and followers.

The authors then proceeded to create a taxonomy of questions on Twitter from their corpora, but first to do so they had to define what a question was, to do this they built upon work by Karttunen, to create five patterns for detecting questions.

By creating the taxonomy the authors wished to articulate generalities that occurred in questions asked on Twitter, and to also build a taxonomy of questions that would benefit from further research, such as information retrieval, visualization, or routing. To classify tweets, a heuristic approach was taken by 5 individuals asked to analyse 100 tweets and classify the tweets, with respect to their authors' purpose in writing them. The results were then refined, and a taxonomy of 9 codes were created to represent types of questions asked on Twitter. Inter-rater agreement was conducted in the form of a Fleiss kappa to calculate inter-rater reliability, the score retrieved was 0.47 and 0.497 which is considered to be of moderate agreement according to Landis and Koch. The attributed this to the king value being high (k=9) indicating that they may expect other see low levels of agreement.

The authors also strived to create alternative taxonomies to dividing the space of microblog questions. In one example they explain how a question can fall within four quadrants of visualization, depending on who the question is aimed at, whether it be targeted to an individual or to their posed questions total their followers at large. Also on the Vertical axis, a tweet could be mesaured by its information need, whether it needed to have immediate tangiable response, or if the information seeker expected a and response however, the insesity that a reply would not be as intense.

Other work such as that by Paul et al. (Sharoda, Hong, & Chi, 2011) has expanded on this subject recently by looking at question and answering in more detail, detailing how a user might receive a higher chance of receiving a response or in some cases a more appropriate response.

2.2.4 Algorithmically Ranking Tweets

Twitter has its own search solution, Twitter search originally began life as Summize, which was acquired by Twitter in 2008. (Twitter Inc., 2008) It offers users the ability to perform keyword search on the Twitter dataset, however at the start of this investigation it would only return results created in the last 7 days. It presented results in reverse choronological order, with a mixture of heavily retweeted content in a prominent position at the top of list.

As well as Twitter, Google and Microsoft (via their Bing search engine) entered the realm of social search. Google fellow Amit Singhai revealed that Google not only ranked individual tweets, but it aslso ranks user's accounts (Talbot, 2010). However, no more data has been released on how it ranked users.

Much speculation has gone into how best to rank users, ranking users by the number of followers, can be perceived to be a bad thing according to Cha et al. (Cha, Haddadi, Benevenuto, & Gummadi, 2010) in their paper Measuring User Influence in Twitter: The Million Follower Fallacy. Influence is being a popular research topic with in microblog research, papers such as that by Bakshy et al. (Bakshy, Hofma, Mason, & Watts, 2011) Identifying 'Influencers' on and by Pal & Counts (Pal & Counts, 2011).

Sean Suchter (Sullivan, 2009) explained how tweets are individually ranked dependant on a number of factors such as estimated authority of the author tweeting, as well as the number of times a tweet had been retweeted and finally the 'freshness' of a tweer, all contributed to how a tweet was ranked in the Bing search engine.

One of the original proposals to calculating influence over the Twitter network was provided by Daniel Tunkelang (Tunkelang, 2009). Whereby a PageRank like

algorithm was constructed allowing users to pass on influence, depending on who they followed.

Mutations of other algorithms such as the HITS algorithm have been modified and modeled on the Twitter network. Weng et al. (Weng, Lim, Jiang, & He, 2010) took a novel approach to ranking users by measuring influence by both topic similarity between users and the link structure of the network into account.

We've seen how Twitter is being used to answer peoples questions whether than be through keyword search, or being part of a conversation. I have also described briefly how several algorithms have attempted to rank tweets.
The temporal nature of microblogs and the web have spurred on much research in the next section where we present research that has attempted to use microblogs for the detection of certain topics and features.

2.2.5 Prediction, Forecasting & Detection

A large body of research has been conducted into how microblogs, can play their part in predicting and detecting trends as well as events. In this section We present research that deals with the forecasting of future events, and one which looks at detecting events quickly. We predict that this has a large part to play in why a tweet may be deemed useful or not.

A paper from Edinburgh University written by Ritterman et al.; (Ritterman, Osborne, & Klein, 2009) Using Prediction Markets and Twitter to Predict a Swine Flu Pandemic. The authors explore their hypothesis that they could extract useful information from social media sources, and by modeling this information they could yield better results than a model constructed with information from prediction markets in isolation.

The authors looked at using internal and external market data, whereby internal market data models the evolving price using previous price movements, and external market data consists of data which is externally observable.

The authors were able to predict the closing price of a prediction market and by doing so were able to explicitly model changes in belief, using data from Twitter.
By using bigram extraction, and historical data to predict changes, the authors were able to achieve a low percentage of error, when predicting prices within the market.
As well as predicating future events, researchers have been looking at how we can best detect new events that appear on Twitter.

Work described earlier in this chapter to do with First Story Detection and Crisis management all play a part in this type of research.

In the next section we look at the temporal aspects of search and how systems have been made to make use of this temporal data.

2.2.6 Temporal Data & Microblogs

> "uncovering patterns of temporal variation on the Web is difficult because human behavior behind the temporal variation is highly unpredictable."
>
> - (Yang & Leskovec, 2011)

One of the interesting dimensions to social media is the temporal aspect, and the ability to detect change of patterns, and the prediction of patters in the future based on previous temporal patterns. There is a growing body of research into discovering more about temporal patterns in microblogs,

Not all of the work carried out surrounding temporal events, has been primarily concerned with detection algorithms.

Work by Yang and Leskovec (Yang & Leskovec, 2011) attempts to uncover the temporal dynamics of online content. The authors were able to create a clustering algorithm, in the attempted to find distinct shapes of time series, the algorithm was tested on two data sets (one contained 580 million tweets, the other 170 million blog posts and news media articles).

The algorithm created by Yang and Leskovec was called K-Spectral Centroid (K-SC), the algorithm effectively finds cluster centroids with a similarity measure defined by the authors.

The paper set out to understand what kinds of temporal features are exhibited by online content, using the datasets mentioned above. As well as this the research aimed to discover how different media sites shape the temporal dynamics of the internet, and what kind of temporal patterns they produce and influence.

To do this the authors examined the data sets, for the Twitter dataset they examined the adoption of hashtags (#something). As well as this they tracked the attention given to various pieces of content via counting the number of mentions over a period of time.

A time series clustering problem was created, and a time series shape similarity metric that was invariant to the total volume and the time of peak activity. Based on the metric the authors developed a novel algorithm for clustering time series, the authors were then able to improve their algorithm by reducing runtime and allowing the algorithm to run over large datasets.

By using their novel algorithm the authors found that the adoption of hashtags in Twitter and propagation of quoted phrases on the web exhibited nearly identical temporal patterns .As well as this the authors state their model allows a 75% accuracy rate to predict which temporal patterns a popularity time series will follow.

The authors suggest the results they were able to create from this work, would have direct application for predicting overall popularity and temporal trends exhibited by online content, as well as finding influential blogs and Twitter users.

Work carried out by Kulkami et al. (Kulkami, Teevan, Svore, & Dumais, 2011) in the paper Understanding Temporal Query Dynamics looks at how queries, their associated documents and query intent changes over time. It is worth noting this

work was not performed on a microblog dataset, but instead on a corpus of web logs from the Bing search engine.

The authors were able to identify several interesting features by which changes to query popularity can be classified. The authors demonstrate the presence of these features, when accompanied by changes in result content, can be a good indicator of change in query intent.

This work has similarities to the work carried out by Yang and Leskovec, as the authors have also identified structures this time not in content authorship, but instead analyzing the structures created from query popularity over time, to identify query intent.

In a paper written by Shamma et al. (Shamma, Kennedy, & Churchill); Tweetgeist: Can the Twitter Timeline Reveal the Structure of Broadcast Events? the authors present two pieces of work surrounding temporal events and social media.

The authors attempts to explore applications for enriching experiences around live visual media events using Twitter to enhance the experience for users of their system. The authors concentrated on two events, one which had already passed, and one which was happening in real time. The authors present a method for segmenting and annotating media using conversational activity from Twitter, for post-event data. They also present methods to aid discovery of current topics of discussion on Twitter and their levels of interest via a real-time feedback display.The paper offers an initial exploration of approaches for applying cues mined from conversation via Twitter, towards enhanced experiences surrounding (visual) media events.

The software (Statler) aims to identify interesting moments from with tweet streams, and is not to purely show overall volume,. It does this by exploring the relationship between the news media and community annotation.

The authors were able to create and display an array of metrics to users of their system, including metrics for 'chatiness' and 'importance'. One of the interesting

methods the authors came up with to detect salient terms involved creating make-shift documents out of temporally co-occurring messages. Using predefined temporal boundaries, (though it suggested a sliding window may be used) to create pseudo documents using all of the messages sent during a given time frame.

Authors are then able to rank terms according to their frequency within these pseudo documents normalized against their overall frequency.

By doing this the authors suggest we can instantly detect temporally salient terms. We are then able to track the change of salient terms over time. By doing this we are able to create cues about the content and structure of an event.

2.3 Summary

In this chapter we have given a description of Twitter, as well as how researchers and industry have tried to create tools to aid people searching over twitter data. One of the biggest takeaways from the literature is that performing search on twitter is a difficult task, not only have we got sheer amounts of data at such speed, but we also have a very noisy dataset with a unique language and set of conventions.

Whilst traditional IR classifies relevancy as something binary. We now know that there are lots of factors which make something relevant to us especially in websearch. In the next chapter we describe a study we ran that looks at what makes tweets useful to people performing microblog search. This will then allow us to make a search engine that targets useful information and tries to display the most useful information to the user.

Chapter 3: What Make a Tweet Useful?

Please note that some of the work described in this chapter was carried out during the author's Masters program and published in their M.Sc. Thesis (Hurlock, Searching Twitter: Extracting Useful Information, 2010).

The data collection stage of this chapter with reference to tweets either being useful or not useful to a user's submitted query was undertaken during the author's M.Sc.

A version of the grounded theory, which is discussed in this chapter, was performed using a single coder during the Author's M.Sc. However, due to having obtained a low kappa score. The Author then performed a more rigorous approach described in this chapter (during their Ph.D.), which was conducted by 2 coders generating codes, and analysis included both coders and a third independent coder to perform the kappa analysis described in this chapter.

All analysis and results described in sections 3.2 and 3.3 were undertaken during the Author's Ph.D.

In this chapter describe an experiment we conducted that led us to find features as to what may make a tweet deemed to be useful, or not to be useful to a user searching a Twitter corpus. Search tasks were based on three of the most common types of search task performed over microblogging datasets. We introduce the experiment, explain the experimental setup, following this with an explanation of the analysis and results.

As described in Chapter 1 of this thesis our work is motivated towards the development of a search service, which provides users with useful information. However, there is as of yet no definition of what factors makes a tweet useful to an information seeker. Reading through the literature there was no mention of usefulness in terms of search.

The closest thing we came to usefulness, was fulfilling an information need. As mentioned in Chapter 1, classically IR takes the view that users are searching to fulfill an information need, Schneiderman et al. (Schneiderman, Byrd, & Croft, 1997)

defines an information need as "The perceived need for information that leads to someone using an information retrieval system in the first place."

Jansen et al. (Jansen, Booth, & Spink, 2007) argue that the Internet search is very different than that of traditional IR practiced and researched. We now know that users are not just performing searches to satisfy information needs, but have other intents whilst searching. For instance in web search we can observe transaction, navigational, informational as well as casual-leisure search (Jansen, Booth, & Spink, 2007) (Wilson & Elsweiler, 2010).

There are millions of communications posted to twitter each day. With 400 million messages sent per day as of 2013 (Twitter, Inc., 2013), there have been numerous attempts to extract clusters of the type of messages sent (Tsur, Littman, & Rappoport, 2013) (Java, Song, Finin, & Tseng, 2007).

In 2009 Pear Analytics released a report stating that 40% of tweets were considered to be mindless babble with another 37.55% being classified as conversational in nature (Pear Analytics, 2009). At this point in time twitter users were only sending 2 million tweets per day (Twitter, Inc., 2010).

We propose that within these millions of messages sent there are tweets with valuable content that may be considered useful to an information seeker. Morris et al. (Morris, Teevan, & Panovich, 2010), as well as Efron and Winget (Efron & Winget, 2010) have looked at behavior of users in social networks, and observed that users do ask their social networks questions, and in-turn receive replies. As well as this, research conducted by Boyd et a.l (Boyd, Golder, & Lotan, 2010) and Java et al. (Java, Song, Finin, & Tseng, 2007) have shown that users share valuable information through posts and links on twitter.

With the knowledge that people seek answers from their social network, and people share information throughout social networks, we infer that social networks such as Twitter, have what may be deemed as valuable information in them.

3.1 Experimental Setup

We know there is a lot of information, and we hypothesis that a certain percentage of this information maybe of use to information seekers, based on the current literature.

We wished to test this theory and to see if we could discover key elements as to what makes tweets useful to information seekers.

To do this we setup an experiment, whereby we asked participants to perform three common search tasks performed over microblog corpuses. These three task were informational search tasks based on those observed by Morris, Teevan & Panovich (Morris, Teevan, & Panovich, 2010).

Using a custom built search engine, we asked them to enter queries related to these tasks, after which participants would rate individual results as either useful or not useful. As well as recording the result as useful or not useful participants are asked to provide a reason as to why they gave the rating for that specific tweet and task.

At the start of the study we gathered demographic data regarding our users, and conducted semi-structured interviews with each participant after they had completed each of the tasks, to discuss their thoughts surrounding the task, as well as their reasoning to marking certain tweets in the way they had. After completing all of the tasks, the study was concluded with a feedback questionnaire and a final and short debrief. In total no user was subjected to the study for longer than one hour.

3.1.1 The Tasks

As previously stated we chose three different types of search task, all informational based on the finding of Morris et al. (Morris, Teevan, & Panovich, 2010) The first task was a temporal monitoring task, the second a subjective choice task and the third a location-sensitive planning tasks. During the experiment, task order was counterbalanced in order to remove any ordering effects.

The first task (temporal monitoring task) involved users to identify interesting information about an on-going event. At the time of the study the most significant culturally relevant event was the BBC Proms.

The second task (subject task) involved users to find information that might help them decide whether to buy the new iPhone. Participants were asked to identify information that might help them to make their decision.

For the third task (location-sensitive planning tasks) participants were asked to identify somewhere nice to eat lunch in London, and identify information that helped them decide where they might go.

3.1.2 System

We created a system that allowed users to rate and comment on what made tweets useful or not useful to them. An overview of the key components and information flow can be seen in Figure 3.1.

A screen shot of the search engine participants used (the study interface) can be seen in Figure 3.2. The interface allows users to enter a keyword search in the top bar, then press search. This would then in-turn grab results via the Twitter Search API.

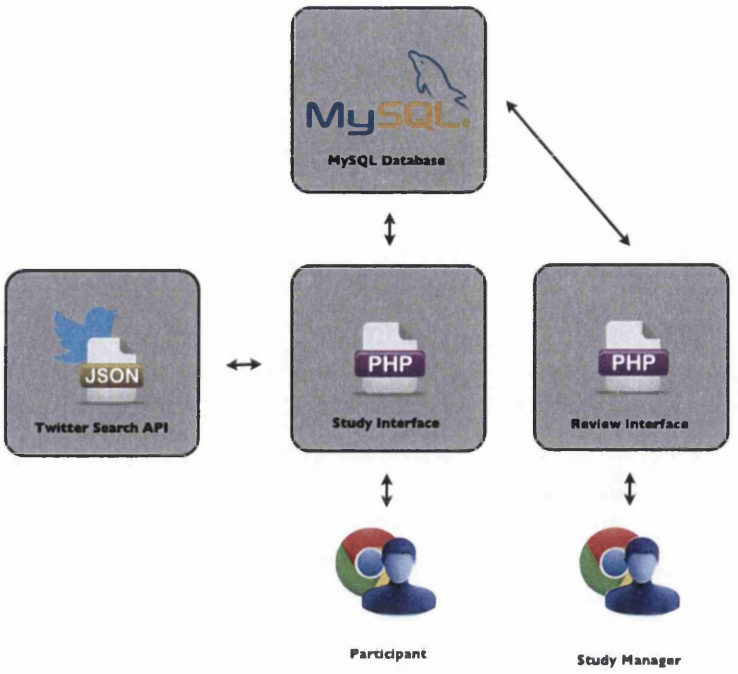


Figure 3.1. A diagram showing the architecture and communication between the different parts of the study.

These results were then displayed below the search bar, with two buttons next to each search result. Allowing the user to then indicate by clicking if the result was either useful or not useful. Once a user had clicked one of these buttons, a text box would appear underneath the result, allowing the user to write their reasoning to why the result was either useful or not useful. Users could then commit their reason and judgment by clicking the corresponding save button.

Whilst the user was searching through the results page, the search engine would also automatically keep searching for more results in the background via an AJAX request at intervals of 15 seconds. It would alert users if it found more results via the bar at the top of the search page this behavior can be seen in Figure 3.2. If users wished to load these results, they could click on the bar, and the new results would appear at the top of the page.

There were two interfaces created for this experiment, the search engine the participants were using, and a researcher's interface. The researchers interface, allowed us as the researcher, to look at all results marked by the user and useful or not useful, as well as the reasoning for the judgment and the search query inputted by the user. This interface would update automatically every few seconds, to allow us to start

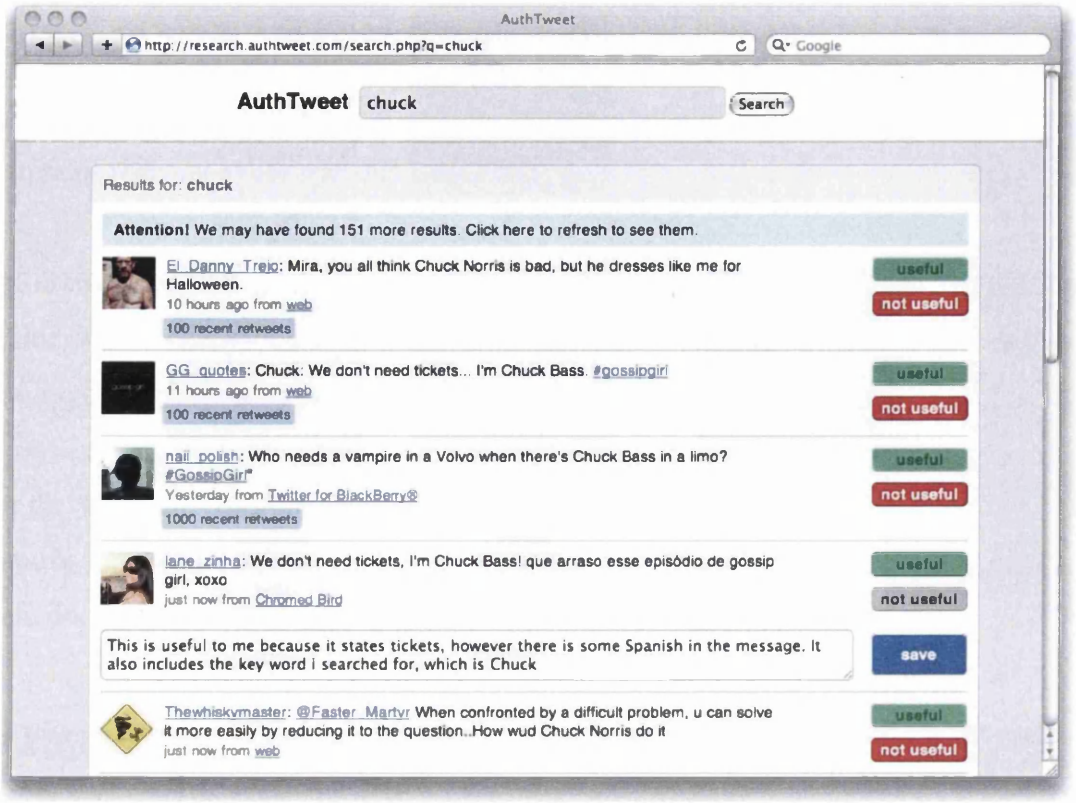


Figure 3.2. The search interface from the user study. Showing a user marking a tweet as a useful, and entering a reason.

thinking of particular questions to ask the user after each task. It also allowed us to show the user what they had marked as useful or not useful as well as their judgment if they wished to reflect on a certain result, or if they stated they made a mistake, by accidentally clicking not-useful rather than useful or visa versa.

3.1.3 The Participants

In total 20 participants were recruited for the study. Participants were recruited internally from within the university via an email sent to both staff and students. We were able to recruit 10 males and 10 females, though this was not intended. Half of the participants were between 20-35 and the other between 36-50, in a roughly normal distribution. 90% of participants held a bachelors degree or higher.

As well as asking participants about their general demographic information, we also asked them about the computer and internet usage. All 20 participants stated that they used the internet everyday, with the majority spending more than 2 hours online per day.

When asked about their twitter usage, 12 of the participants stated that they had used Twitter, 30% on a regular basis and only two participants stated that they have attempted to search Twitter directly.

It is important to note it was not a condition of our study that participants to have used twitter before. The reasoning behind this was that we wished to find useful tweets that could be returned on any search system. Google (Search Engine Land, 2011) and Bing (Search Engine Land, 2013) used to provide tweets integrated within their search results, but have since removed this service. Also by having a mix of familiarity with Twitter allowed for a broader perspective on what constitutes useful social media content.

3.2 Analysis

In total participants rated 496 tweets, of which 482 were unique. Of the original 496 ratings, 52% were considered to be useful to participants. After splitting the data from the tasks into two sets (useful and not useful), an inductive grounded theory (Glaser & Strauss, 2009) approach was used to reveal commonalities in the comments made about the tweets.

By using a grounded theory approach we were able to establish a systematic procedure for identifying common topics and themes in the open texts gathered when a user either marked a tweet as either useful or not useful.

Pieces of text were given 'codes' that represented their meaning. These codes we then grouped into themes, and used to produce underlying theories about the data. The rigor of our approach is detailed in the following paragraphs.

Analysis began with both coders evaluating an initial set of 100 useful and not-useful 'tweet+response' pairs independently. Both coders then met and compared the codes created thus far. This allowed the coders to both reflect on the dataset, and broaden our perspectives of the dataset and possible codes.

Each coder then continued to code the remaining tweets independently. Coders concluded the inductive coding by collating all the codes we had each created together and using a white-board affinity diagramming approach, commonly used to organize unstructured sets of ideas and concepts, to begin identifying relationships between themes and codes in the tweet+response pairs, until the diagram stabilized. This process was then repeated for the not-useful tweets.

From the original set of more than 30 proposed codes, coders settled on 16 codes for useful tweets, and were able to agree on 6 categories of 2-4 codes each. Similarly, coders reduced the set of proposed not-useful codes and identified 17 codes that also fell into 6 categories of 2-5 each.

To validate these codes, Cohen's kappa (Cohen, 1960) was used to assess the inter-rater reliability. We achieved a high kappa score of 0.85 (Almost perfect agreement according to Landis and Koch (Landis & Koch, 1977)) for the not-useful data set, as shown in Table 3.1. To further validate our results we introduced another member of our research team who was independent to the work being carried out as an untrained coder to the analysis. The independent judge was provided with a set of codes and definitions. Table 3.1 shows the Cohen scores achieved between all investigators, and that together the three coders achieved a Fleiss's kappa (Fleiss, 1971) of 0.73 for the not-useful tweets.

At first, as shown in Table 3.1, coders did not achieve such high scores with the 'useful' tweets, even between the two authors of the paper. Due to this difference in scores, we revisited the tweets and codes, discussed in our findings, and sought to discover where the source of disagreement lay.

From our investigations, we discovered that while the not-useful tweets typically had a single striking reason to be declared so, the useful tweets often had two or three valuable features.

Coder 1 observed an average of 2.14 codes per tweet-response pair in the useful tweets data set, with a range of 4 (Max: 5; Min: 1; STD: 0.90). Coder 2 observed an

average of 2.34 codes per tweet+response pair, with a range of 3 (Max: 4; Min: 1; STD: 0.92). For tweets deemed to be not useful, we saw a much lower average of 1.18, with a range of 2 (Max: 3; Min: 1; STD: 0.44).

As both Cohen and Fleiss analyses are performed when a single code is applied to a piece of text, we had originally asked the coders to choose 'the most appropriate code' for the tweet+response pair. Table 3.1 shows how the investigators easily applied different codes to the same tweet+response pair. Consequently, we sought to evaluate our codes using an analysis method that was suitable for multi-coding individual tweets. We performed a multi-coder, multi-coded kappa analysis detailed by Harris and Burke (Harris & Burke, 2005), and achieved a score of 0.73 between the two coders, which is strong 'Substantial Agreement' according to Landis, and Koch (Landis & Koch, 1977). This high score suggests that our original use of Cohen's kappa was indeed inappropriate. With our independent untrained validating judge, we also achieved multi-coded kappa score of 0.62; still a 'Substantial Agreement', and good given the high variability associated with multiple coding.

3.3 Results

In this section we discuss the results in detail regarding to the reasons why tweets were deemed to be either useful or not useful to an information seeker. Tables 3.2 and 3.2 give us an overview of the codes, grouped by category, that were derived from the data, for both the useful and not-useful collections, respectively.

Useful Tweets Data Set				
	Coder 1	Coder 2	Independent Coder	Fleiss' Kappa
Coder 1	-	0.5065	0.5097	
Coder 2	0.5065	-	0.4607	0.4868
Independent Coder	0.5097	0.4607	-	
Not Useful Tweets Data Set				
	Coder 1	Coder 2	Independent Coder	Fleiss' Kappa
Coder 1	-	0.8585	0.659	
Coder 2	0.8585	-	0.6856	0.7331
Independent Coder	0.659	0.6856	-	

Table 3.1. Showing Cohen's Kappa scores between multiple coders for both the Useful and Not Useful data sets. Also included is the Fleiss' Kappa score for each data set for the agreement between all three coders.

We saw four key reasons where the content of the tweet was directly useful. Some contained facts (e.g. times or prices) or increasingly common knowledge (e.g. problems with the iPhone). Others contained direct recommendations, or relayed insights from personal experiences. We also saw two types of tweets that the user found to be amenable, ones that were funny and ones that shared the searcher's perspective (e.g. Apple products are good or bad). We also saw two codes that focused on whether tweets were geographically or temporally still relevant (e.g. tweets in British prices). We also observed a key theme of trust, where users reported approving of trusted twitter accounts and recognizing trustable avatars for those accounts. Also, links to authoritative or trustworthy websites were frequently recognized. Other links were also important, whether they provided more detailed information, rich media, or services (e.g. buying tickets).

There were also five key reasons that the content of tweets was not useful for the searcher. First tweets were frequently vague or introspective (for the author), or were quite directly not relevant by topic. While some tweets showed potential, it was easy for tweets to be too technical for the reader (containing jargon) or to contain errors (e.g. malformed URLs). There were 3 other reasons for tweets to be badly constructed: containing dead links, spam-style content, and being in a foreign language.

It was important for tweets to be temporally and geographically relevant, many tweets were deemed as not-useful because they were not current and about irrelevant locations. Similarly, non-trust was an issue, where users were not happy with some pieces of information coming from non-authoritative sources, and being linked to dubious websites. Further, not-useful tweets were often repeated content, or part of a conversation that would only be useful as a whole.

There were also three more subjective factors of not-useful tweets, including users disagreeing with the tweets (e.g. being pro or anti Apple), or not finding them funny.

In Tweet Content		T1	T2	T3
Experience	Someone reporting a personal experience, but not necessarily suggestion / direction.	15	12	13
Direct Recommendation	Someone making a direct recommendation, but not necessarily relaying a personal experience.	3	3	20
Social Knowledge	Containing information that is spreading socially, or becoming general knowledge.	7	6	6
Specific Information	Where facts are listed directly in tweets e.g. prices, times etc.	51	10	47
Reflection on Tweet				
Entertaining	The reader finds them amusing.	1	3	2
Shared Sentiment	The reader agrees with the author of the tweet.	1	2	1
Relevant				
Time	The time is current.	14	0	2
Location	The location is relevant to the query.	6	1	40
Trust				
Trusted Author	The twitter account has a reputation / following.	3	2	6
Trusted Avatar	The visual appearance cultivates trust.	2	0	2
Trusted Link	A link to a trustworthy recognizable domain.	14	1	7
Links				
Actionable Link	The user can perform a transaction by using the link (heavily dependent on trust).	9	0	0
Media Link	The link is to rich multimedia content.	9	0	0
Useful Link	The link provides valuable information content, e.g. authoritative information, educated reviews, and discussions.	61	30	43
Meta Tweet				
Retweeted Lots	Its information that others have passed on lots.	4	0	4
Conversation	It is part of a series of tweets, and they all need to be useful.	1	4	4

Table 3.2. The 16 codes and the 6 categories extracted from responses and tweet pairs from the useful tweets. Further, columns 3-5 show how frequently each was associated with the temporal (T1), subjective (T2) and location-sensitive (T3) tasks.

Tweet Content		T1	T2	T3
No Information	Absence of anything, event factual points.	16	14	12
Introspective	Personal content and personal thoughts for no social benefit.	5	5	8
Off Topic	Result not related to the query given / TF-IDF irrelevant	27	21	18
Too Technical	The content requires specific domain knowledge the reader doesn't possess.	1	2	2
Poorly Constructed	Tweets that may have grammatical/spelling errors, or malformed URLs.	3	2	3
Bad Tweets				
SPAM	Irrelevant or inappropriate messages.	0	17	2
Wrong Language	Messages sent in a foreign language of that to the reader.	3	2	1
Dead Link	A URL which does not work i.e. 404	2	4	3
Not Relevant				
Time	Out of date content.	0	1	1
Location	Wrong geographic location.	2	7	2
Trust				
Un-trusted Author	An author the reader feels at un-eased by or suspicious of.	4	7	1
Un-trusted Link	A link the reader feels is suspicious	4	7	2
Subjective				
Perspective Oriented	A tweet that is perspective centric, meaning the author is providing their views or projecting an attitude on a subject matter or to a subject/reader.	2	3	2
Disagree with Tweet	A conflict of agreement between the reader and the author	2	2	1
Not Funny	A tweet that is aimed to be humorous, which the reader does not feel is humorous.	1	1	1
Meta Tweet				
QnA	Part of a conversation, reader desires the whole conversation, not just the question or the answer, but both the question and answer	2	4	9
Repeated	Content the reader has seen before	3	7	1

Table 3.3. The 17 codes, in 6 categories, extracted from responses and tweet pairs from the not-useful tweets. Further, columns 3-5 show how frequently each was associated with the temporal (T1), subjective (T2) and location-sensitive (T3) tasks.

3.3.1 Analysis by Task

In the next few sections we provide an analysis for each specific search task. Tables 3.2 and 3.3 include counts for how frequently each code was applied to tweet+response pairs for each task.

3.3.1.1 Temporal Search

For the temporal search task, useful and trusted links along with specific information, played main factors in deciding if a tweet was useful for that task. We also saw how

other types of links, including media, were also frequent for the first task. The increased popularity of the media link code may have been influenced by the broadcast of the BBC Proms over the Internet. Media links, did not account for other tweets being regarded as useful for other tasks.

3.3.1.2 Subjective Search

For the subjective task, we were able to observe that experience with or of the subject matter was important to the information seekers. We also see two very interesting codes appear in this task, which are able to compliment each other, the first being shared sentiment, and secondly entertaining. Both of these codes are subjective in nature, which could be expected of a subjective task. Useful links and experience were also played an important role in this task. Many participants found this task frustrating due to the amount of non-useful tweets; many of them were marked as SPAM or untrustworthy.

3.3.1.3 Location Sensitive Search

In the third (location-sensitive) task, we again see a high dependency on specific and useful information. However for this task, specific information played a more important role. As suspected we also see location sensitivity as an important factor, dominating this task with 85% of reasons to why location sensitivity is useful being allocated to this task. In this task, we see that trust, in the form of avatars and authors played an important role, with 2 tweet+response pairs being coded as useful because of the participant trusting the avatar, and a further 6 being coded as trusted author. Further, we see the introduction of direct recommendation and experience playing a part in why a participant found a tweet useful. Perhaps indicating a need for knowledge of first hand experience from someone who has been to a lunch venue in London, rather than a commercial entity trying to sell an experience or product. Location-sensitive task, averaged at 2.75. No participant rated a tweet with a score of 5 (very relevant). This scale was based on a Likert scale (Likert, 1932). 20% of participants, however, gave a score of 0 (not relevant) during the second subjective task, but not in the temporal and location-sensitive tasks.

3.3.2 Common Patterns

As well as statistical analysis of the codes we were able to pick up on structural traits of tweets. Some of the structures that we were able to extract combined several of our

codes combined together to make a structure. One in particular, which we called a teaser, combined codes for specific information and a link, which accounted for 22% of the useful tweet+response pairs. Another 13% were coded holding both specific information and location codes, which we attributed mainly to the location-based task. Another structural concept we came across was actually a code QnA which is where a user could only see part of a question whether that be the question itself or an answer to a question, but could not see both parts, or multiple answers. The QnA code was found in 6% of the not-useful dataset and highlights the need for returning responses to question-tweets returned by a search.

Twitter itself has tackled some of these concepts when browsing its website. For instance the embedding of images and some videos in its new layout. As well as the 'in reply to' feature shown when browsing the site (Williams, 2010). These features have failed to make it over to Twitter's search service.

3.3.3 Additional Findings

We also found additional evidence for identifying tweets from authors that people may recognize. In lieu of identifying tweets that are socially connected to the searcher, our analysis suggests that authority measures, such as TunkRank[1] and Klout[2], could also be used to assess estimated trustworthiness.

We were also surprised to see that some codes, such as 'Retweeted lots', did not feature as highly as we had expected based on emphasis of previous work on retweets. With just under half of participants stating they have not used Twitter, and only 30% stating they use it regularly, we suspect that unfamiliarity with Twitter specific features may be a reason.

Although most of the post-task interviews simply elaborated on the points noted by participants during the study, a few additional factors were identified. One potentially interesting additional factor was the impact of a tweeter's avatar. Many users suggested that avatars were a factor in choosing whether a tweet was trustworthy or

[1] http://www.tunkrank.com

[2] http://www.klout.com/

not; most stating that they like to see faces of individuals. Several participants stated that they thought they would be able to tell if a tweeter had similar preferences to them by just looking at their avatar. One participant, for example, said: "Why would a baby give me a free phone?? Automatically suspect a con or a virus!" This suggests that both the type and presence of an avatar have an affect on the trustworthiness of tweets. On discussing the importance of trust, another participant said "... Also think I know this tweeter - a friend of a friend - so might be inclined to try the restaurant anyway!" These findings about trust echo the principles of Aardvark's social network routing efforts (Horton & Chilton, 2010), but the emphasis on visual avatar judgments is important to note for future systems.

When asked if users were able to guess where authors were when they tweeted, or when they tweeted, most participants stated they were not aware of these factors, unless some specifically said 'I am in...' It appeared, through discussions with participants, that metadata played a very small role in their search experience. This may be a factor of the way results are displayed in Twitter, but could imply that metadata is more useful for the algorithms than the searcher.

In regards to query size, participants also mentioned frustration when searching, noting that longer queries returned much fewer results, or no results at all. Users noted that shorter queries, using one or two general terms were much more productive. This is likely due to the short limited size of tweets. Social search user interfaces may wish to encourage shorter, more general queries, but will have to work harder to identify the implied contexts associated with them.

Whilst we are not strictly looking at relevance, research by Spink et al. (Spink, Greisdorf, & Bateman, 1998) suggest that relevance is a very multidimensional which our findings agree with. However, in the article goes on to state "so many factors have been suggested as affecting relevance judgments that it is not possible to list them all here." , though they do list 80 (which is a subset).

In the article the authors state "The measures of usefulness, ... and satisfaction measure other important factors that users may employ in making relevance

judgments and are sometimes used in research as an alternative way to define and measure relevance". It is very important to note that at no point did we define exactly what we meant by "useful" or "not useful" to user, we left this open to interpretation. Even though we left this definition open, we see many traits of relevance in usefulness, and loosely agree with this statement made by Spink et al, due to similarities found within the literature and research.

In the article Spink et al. explore 4 studies looking at relevance judgments, and how different levels of relevance presented to users effect their information seeking process.

One of the key findings was that IR and relevance researchers should question the assumption that highly and partially relevant items have the same utility for users.

Their findings suggest that both partially relevant and highly relevant items may have a potentially important role to play in the evolution of solving a user's information problem. They suggest that partially relevant results may prove a crucial role in providing users with new information and directions that may lead them through further stages of their information seeking process toward a possible resolution to their information problem and fulfilling their information need.

Whilst we did not formally record this, we saw user's information seeking behavior change based on the information they had observed within the search results. One example of this was when a participant found a specific restaurant and location based after searching with very generalized queries for sometime, their behavior changed that they then started formulating more queries specifically targeting the restaurant name and also the general location of the restaurant.

It is hard to predict what the user's domain knowledge and intent is before them telling you, and even harder to programmatically represent this. It would therefore be hard to dynamically serve 'less' or 'more' relevant results to the user. More importantly the question is can we tailor an experience to a user in which we either

fulfill their information need or provide serendipitous results to them depending on the type of search they feel like performing (type of interaction) at any given time.

It is worth noting that there are some items in our codes for useful tweets that are mirrored in our not-useful codes. Further exploration of this relationship would be both interesting and beneficial, and will be discussed in Chapter 5. This analysis helps to measure the influence of different features on a single tweet, when it contains both useful and non-useful features, as well as comparing codes against each other in terms of influence.

Although the majority of our codes can be objectively identified, there were a few features that were subjective or perspective-oriented. One clear example was whether the searcher and the tweet-author were both pro or anti companies like Apple or Microsoft. Such perspective-oriented examples were clearly seen between codes 'Entertaining' (in tweet content from Table 3.2) and 'Not Funny' (subjective from Table 3.3). Perspective oriented presents us with the challenge of trying to create a system that learns about a users preference, and trying to tailor results to that specific user.

In an article written by Barry and Schamber (Barry & Schamber, 1998) the authors attempt to understand the behaviors of end-users by focusing on the values or criteria they employ in making relevance judgments or decisions about whether to obtain and use information. They do this by comparing two user criteria studies that are similar in terms of methodologies. However, the types of users, information formats, sources and environment differ. The authors compared and contrasted user criteria for relevance evaluation, producing a list of common criterion categories, which were decided to make information relevant to users. The codes and categories which we have produced bear some resemblance to those created by Barry and Schamber, such as Currency (The extent to which information is current, recent, timely, up-to-date), Quality of Sources (The extent to which general standards of quality or specific qualities can be assumed based on the source providing the information; source is reputable,trusted, expert). There are also some cases where we saw some of the non common criterion categories, for instance Barry saw 'Relationship with author', and

when performing the study one participant mentioned they knew one of the authors who had returned a result, ultimately this was coded as trusted author, as the user knew the author personally. Another example of where we saw similarities, was with Schambers categories where Schamber identified geographic proximity, this was possible to due with Schamber's tasks involving weather information. In both our useful and not useful codes, we highlighted location / geographic proximity, as if a tweet was deemed to be about another geographic location, especially in the location based task, it was seen as not useful.

Barry and Schamber conclude that some codes which are not deemed to be 'common' may appear due to the differences in situational context, and research task requirements, however, it is not due to inherent differences in evaluations behaviors of respondents.

3.4 Summary

The primary contributions of the work carried out in this chapter has been the production of a set of reasons as to why information seekers find tweets to be usefulness or not useful, when performing the three most common type of search tasks carried out over a microblog corpus.

By using qualitative and quantitative data analysis, we have been able to produce two lists that identify the traits of tweets that provide useful information, and of course those that do not. We have also observed and discussed how certain combinations of codes can enable a tweet to be deemed useful. It was also observed that certain codes were deemed to be more important for certain types of search task.

Now that we have discovered reasons as to what constitutes a useful and non-useful tweet, it will allow us to create a search system that allows us to analyze tweets for these features and combinations of these features. In the hope of allowing us to improve the search experience for information seekers, providing more useful results to them. The majority of the features we found are objective and easily identifiable characteristics, whilst some are subjective in their nature, and are not so easily identifiable.

In the next chapter we present a system which allows us to extract and target the features we have identified in this chapter, allowing us to build an information retrieval system that allows us to search through a microblogging corpus and filter tweets depending on their usefulness.

Chapter 4: Building a Search Engine

4.1 Introduction

In the previous chapter we described a study we performed which allowed us to find reasons as to what made tweets either useful or not useful to people performing popular search tasks over a microblog corpus.

Now that we have identified these reasons we wish to build a system that is able to identify these attributes in tweets, and allows us to search through a corpus of tweets we have collected to return useful results. In this chapter we discuss the architectural decisions we have made to allow us to create a system which allows us to identify these attributes and to retrieve them in a system which a suitable search time frame.

We start by describing the datasets available, followed by the possible frameworks available, finally discussing the chosen implementation.

4.2 Datasets

For the purpose of this Ph.D. it would be impossible and unrealistic for us to obtain a copy of all tweets ever sent and currently being sent (See Chapter 2 to see sheer volume). So we have had to rely on test corpuses to test our system. In this section I describe the corpuses, as well as the advantages and disadvantages of each.

It is worth noting at time of writing, you can no longer obtain full sets of these corpuses due to Twitter's terms of service. You may obtain a list of tweet ids, for each corpus, but if a tweet has been deleted, then that tweet can no longer be obtained via whilst sticking to the terms of service.

4.2.1 SNAP

The Standford SNAP twitter7 corpus[1] consists of 476million tweets from 17million users, which were collected between June and December 2009. It is estimated that this was 20-30% of all tweets sent via the service at time of collection.

[1] http://snap.stanford.edu/data/twitter7.html

Whilst this collection is supposedly represents a large proportion of all tweets sent, the collection only contains the username as well as the message, no meta data is included in the corpus. Whilst this makes it very interesting to perform text analysis on, it does not allow us to perform all the tasks we wish to on the data. As well as this is data from early on in Twitter's life time, the conventions used are not fully up to date with current trends.

4.2.2 Edinburgh Corpus

Researchers Miles Osborne and Sasa Petrovic at Edinburgh University collected the Edinburgh Corpus. At time of publication (Petrovic, Osborne, & Lavrenko, The Edinburgh Twitter Corpus, 2010) the corpus consisted of 97 million tweets covering November 2009 to Feburary 2010. Again like the SNAP corpus, the Edinburgh corpus did not offer all the meta data included in a tweet, and was again an 'old corpus'. However, it did provide a little more information than the SNAP corpus. The Edinburgh corpus included a timestamp when the tweet was sent, an anonymous username for the author of the tweet, the tweet message as well as client information (what application was used to post the tweet).

4.2.3 TREC

Possibly one of the most complete corpuses, was that provided by TREC. This corpus was setup after Twitter prevented users from sharing complete corpuses which each other. This corpus was created, to allow researchers to compare differences between corpuses.

The TREC corpus consisted of approximately 240 million tweets, which were collected between February and March 2013. The way these tweets are obtained is via the TREC corpus tool, which takes tweet ids as inputs, then queries twitter via the API to see if the tweet is still available, and if it is, it then downloads the JSON for that tweet. However, due to rate limiting only 150 tweets maybe downloaded per hour per account. So this takes considerable time and resource to obtain a full corpus. There are two versions of the tool. One which downloads the JSON for the tweet, and one which HTML scrapes twitter, which is considerably faster as it not rate limited by twitters API, however it does not obtain all meta data for the tweet.

Unlike the SNAP and Edinburgh corpuses, which were full corpuses you could download in one block. The TREC corpus 'degrades' over time. This has been documented by Ounis et al. (Ounis, MacDonald, Lin, & Soboroff, 2011). The reason for this is, as you are fetching tweets on the fly, the corpus is always changing, for instance if a user deletes their account, then all the tweets with that user are removed from Twitter, and are no longer obtainable. Also if a user changes anything to do with their profile, e.g. their name, avatar, bio etc. then this is what will be saved at time of scrape, so if two people run the same scrape at different time frames, then they are likely to have two different datasets.

4.2.4 Custom Twitter Data

The way in which all of previous corpuses were generated, were by utilizing Twitter's data streams. Twitter allows users to see a glimpse of data running through their service, via something commonly called the Spritzer Stream[1]. This is a constant stream which returns a small random sample of all public status.

As well as this there is also the Firehose[2]. The Firehose, returns all public statuses, few applications have access to this stream and access has to be granted via twitter, realistically it is a very expensive task to be able to process and store tweets that are delivered through the firehose[3]. Ideally our system would utilize the firehose allowing indexing and retrieval of all tweets, however we do not have access to the firehose, so have instead used the spritzer stream through out this project.

We have created scripts that allow us to save the output of the spritzer stream, in a manageable format. As well as the spritzer stream, we have generated scripts that allow us to query the search API at intervals and save any output that is returned for any list of given queries.

---

[1] https://dev.twitter.com/docs/api/1.1/get/statuses/sample

[2] https://dev.twitter.com/docs/api/1.1/get/statuses/firehose

[3] https://dev.twitter.com/discussions/2752

4.2.5 GNIP & DataSift

As well as storing our own copies of data we have recorded from either the streaming API or search API, Twitter also allows you to buy data from one of its resellers. GNIP and DataSift are services which allow you to 'buy tweets'. Each company allows you to either buy realtime tweets or historic tweets.

In terms of real-time tweets, GNIP allows you to buy a sample stream called the Decahose – 10% of the firehose, or allows you to use one of their products to help you filter and buy relevant tweets. DataSift on the other hand allows you buy all tweets that have been filtered through their system, using their custom filtering language.

In terms of Historics, GNIP offers tools allowing users to search via filters for specific features and/or keywords and for historical tweets, DataSift offers a similar service. However, neither provider allow you to download blocks of tweets in terms of getting all tweets between date and time x and date and time y.

4.2.6 Choosing a DataSet

When we originally started this project we started to build the system using DatiaSift,to identify tweets that contained attributes for the codes we found in Chapter 3. At the time, it was free to use DataSift with twitter data, however in late 2011 DataSift changed this, so that we would have to be charged to use these codes. At this point we abandoned using DataSift due to the cost of building a system in this way would be too expensive.

After this change, we started to use the SNAP corpus for testing some of the codes. However, we quickly found that even though the SNAP corpus was great for testing some of the codes, it was not a good corpus to utilize for other codes as it does not include all of the meta data. As well as this, due to the size of the corpus, it took a considerable amount of time to index the whole corpus on the resources available to us.

After trying both the DataSift and SNAP corpus, we decided to generate our own corpuses for testing. We discuss one of these corpuses in detail in the next Chapter. The corpuses we generated were created using custom scripts that scraped both the Twitter Search API and the Spritzer Stream.

4.3 Processing

One of the biggest questions we had to quickly resolve with this project was how do we processes all of this data. Do we attempt to process the data on the fly in near-realtime, to do we opt for a batch processing approach. At first when we were using the DataSift platform to filter tweets, we were hoping for a near real-time processing approach. However, ultimately we ended up with a batch processing approach-utilizing Hadoop. We give an overview of some of the frameworks and technologies we encountered and why advantages and disadvantages of both in this section.

4.3.1 Hadoop

Hadoop is an Apache project, it is an opensource framework which deals with the storage and processing of large data-sets. Hadoop processes data in a batch. It was created by Doug Cutting, and is an open-source implementation of the map-reduce framework that was described in the 2004 paper MapReduce: Simplified Data Processing on Large Clusters by Dean and Ghemawat (Jeffrey & Ghemawat, 2008).

Hadoop is probably the most popularly used Batch Processing system for processing 'Big Data', with more than half of the Fortune 50 companies using Hadoop (Noyes, 2014). Companies such as Yahoo and Facebook, use Hadoop to enhance their search services. In 2010, facebook claimed that they had the largest Hadoop cluster in the world with 21 Petabytes of storage, however by June 2012, this had grown to 100 PBs and as of November 2012, they annoced that the data gathered in the warehouse grow by roughly half a PB per day. As we can see it is suitable for processing the amount of data we are likely to encounter, and proves that it is scalable to the desired size, however this would require substantial capital investment.

Due to Hadoop's popularity, it is a well documented project, and has many branches and side projects created which have enhanced the project as a whole, we discuss some of these projects in the Retrieval and Storage section of this Chapter.

The main disadvantage of Hadoop is that is unable to process data in real time/streaming data. It would be ideal for a search service to be able to take a stream, then do all the processing on the fly. We discuss some solutions that are able to do this later in this section.

4.3.2 HPC Wales

During the late stages of the project, we were introduced to HPC Wales. HPC Wales is an ERDF project, which hopes to allow businesses and researchers access to high performance computing resources.

HPC Wales allows researchers and businesses in the convergence area to access their computing resources at little to no cost. HPC Wales does not provide a framework such as hadoop or any libraries to make use of the number of cores available to the customer. Instead any code deployed to the cluster must be optimized to utilize all cores that are available.

Due to having programmed all the appropriate codes to run on a hadoop architecture and with third party libraries, it was seen as unnecessary work at the point at which HPC was introduced to us, as it would require us to reprogram the codes to utilize the cluster appropriately. That coupled with the guaranteed availability and limited access to the machines meant we chose not to use these resources.

4.3.3 S4

The S4 framework is an Apache incubator project that allows for developers to create programs which deal with continuous unbounded streams of data. It is designed to work in a distributed, scalable, fault tolerant architecture.

S4 was designed to fill the gap between proprietary stream based frameworks and batch-oriented open source platforms (such as hadoop). Like hadoop, s4 is primarily written in Java. At time of writing version 0.6.0(The last commit at time of writing was October 2013.) has been released, and it due to the infancy (Apache project since

September 2011) and lack of documentation with the project that we decided not to pursue using S4.

However, we know that S4 has been used by Yahoo for processing of search queries (Neumeyer, Robbins, Nair, & Kesari, 2010).

4.3.4 Storm

Storm is another Apache incubator project that is primarily concerned with the distributed and fault-tolerant real-time computation of stream data. It is very similar to S4 and is actually used by Twitter. On the Storm website they compare what they are doing with stream processing, as what Hadoop did for batch processing.

One of the biggest advantages of Storm is that it allows any programming language to be used via a Thrift definition, that allows communication over a JSON-based protocol.

There were two main disadvantages to us potentially using storm, like S4 it is still very much in its infancy, and like S4 it has little documentation, this meant if we were to hit any problems, we could be stuck and have no where to turn to.

Based on these options we decided to implement a hadoop to process our data. As was the most stable, well documented processing option available at time of conducting this project. It allowed us the freedom to do some interesting analysis of tweets, as well as not binding us to any one language. We will discuss the hadoop architecture we have employed further in this chapter.

4.4 Storage and Retrieval Engines

Now that we have discussed how we are to process the data, we must deal with the problem of how we store the data and how once we have stored it how we retrieve it with a reasonable response time. In this section we discuss appropriate storage systems, as well as retrieval engines, giving advantages and disadvantages of each solution. We start with HDFS which is used by Hadoop for processing the data in its system, then move on to how we are to retrieve this data originally inserted into HDFS.

4.4.1 HDFS

Hadoop Distributed File System or HDFS is the file system that hadoop uses. It is a distributed file system which is designed to run on commodity hardware. HDFS is based upon the Google File System (GFS), it is highly fault-tolerant. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

One of the main advantages of HDFS is that it enables streaming access to file system data, it was originally built as part of the Apache Nutch web search engine project, however was moved from Nutch to the Hadoop project.[1]

The Apache Nutch project is tasked with building a highly extensible and scalable open source web crawler, stemming from the Apache Lucene project.[2]

4.4.2 Hive

Hive is an Apache project concented with offering software facilites for querying and managing large datasets which reside in distributed storage such as HDFS. One of the biggest advantages to use about hive what that it offers mechanisms not only to store the large datasets we need to work with, but it also offers a declarative SQL like language for performing queries on the data in its store. This language is called HIveQL (Hive Query Language). As well as offering the usual query like operations, Hive also allows users to inject custom mapper and reducers when it is inconvient or inefficient to express this logic in HiveQL.

Hive offers many desirable features for our system, allowing for advance querying and data manipulation. However, when we tested hive the iob initialization time was unsatisfactory. Taking five seconds just to run a select query on a hive table with only one row.

---

[1] http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

[2] https://nutch.apache.org/#What+is+Apache+Nutch%3F

Whilst this would acceptable if we were just performing one pass along the data, this is not the case. Due to some of the codes, we perform multiple passes over existing data, to check SPAM scores as well as URL checking.

4.4.3 MySQL

MySQL is the world's most popular open source database. MySQL is a relational database management system. It also offers full-text search and indexing. As well as this a variant of MySQL called MySQL Cluster also offers distributed storage and replication of data.

When originally choosing a storage engine we looked at using MySQL. However when performing select queries from our database retrieval times were unsatisfactory, again like Hive, retrieval was taking a couple of seconds to minutes depending on the query being run. This was an unsatisfactory amount of time. So we searched for other retrieval engines.

4.4.4 Pig

Pig is an Apache project that is concerned with allowing users to write high-level map reduce programs used with Hadoop. The programs are written in a language called Pig Latin, it resembles a SQL like language, however, it can also be extended using user defined functions, alloing users to write java, python, javascript, ruby or groovy programs tand call them directly from within Pig. Also unlike SQL which is a declaritive language, Pig Latin is a procedural language.

The main advantages of Pig was that it allowed easy creation of map reduce jobs, in a well documented and logical language. However, possibly its biggest advantage was where a suitable Pig Latin function could not found/existed, we were able to program the functionality in another language such as python, and import that functionality via a user defined function.

We programmed some functionality in Pig, however found the same issues arose as that with Hive. The initialization time was considerable and could not be used for on the fly querying of existing data.

4.4.5 Lucene

Lucene is an apache project which develops open-source search software including Lucene Core, Solr, Open Relevance Project and PyLucene. The project offers high performance search services, advance tokenization and analysis of text, as well as implementations of many common search algorithms.

Solr the enterprise search platform built onto of Lucene allows full-text search, near real-time indexing, as well as allowing for scalable fault tolerant and distributed indexing as well as load balancing. All of these qualities are very desirable features when building a system such as this.

Solr is an enterprise search platform and as such there is a lot of through documentation, which is ideal. However, it is written in Java, and whilst this may seem as an advantage as it is cross platform compatible. It meant if any customization such as result boosting or custom tokenization of strings would have to be done in Java, or written in another language with a Java wrapper.

As we had not written any Java code for a few years, we wished to see if there were any other frameworks which supported a language that we felt more comfortable.

4.4.6 Katta

Katta is distributed search framework. It allows indexes to be imported straight from either Lucene or Hadoop. Which is advantageous, as we have chosen to use Hadoop to process our data. The Figure 4.1 describes how Katta integrates with Hadoop.

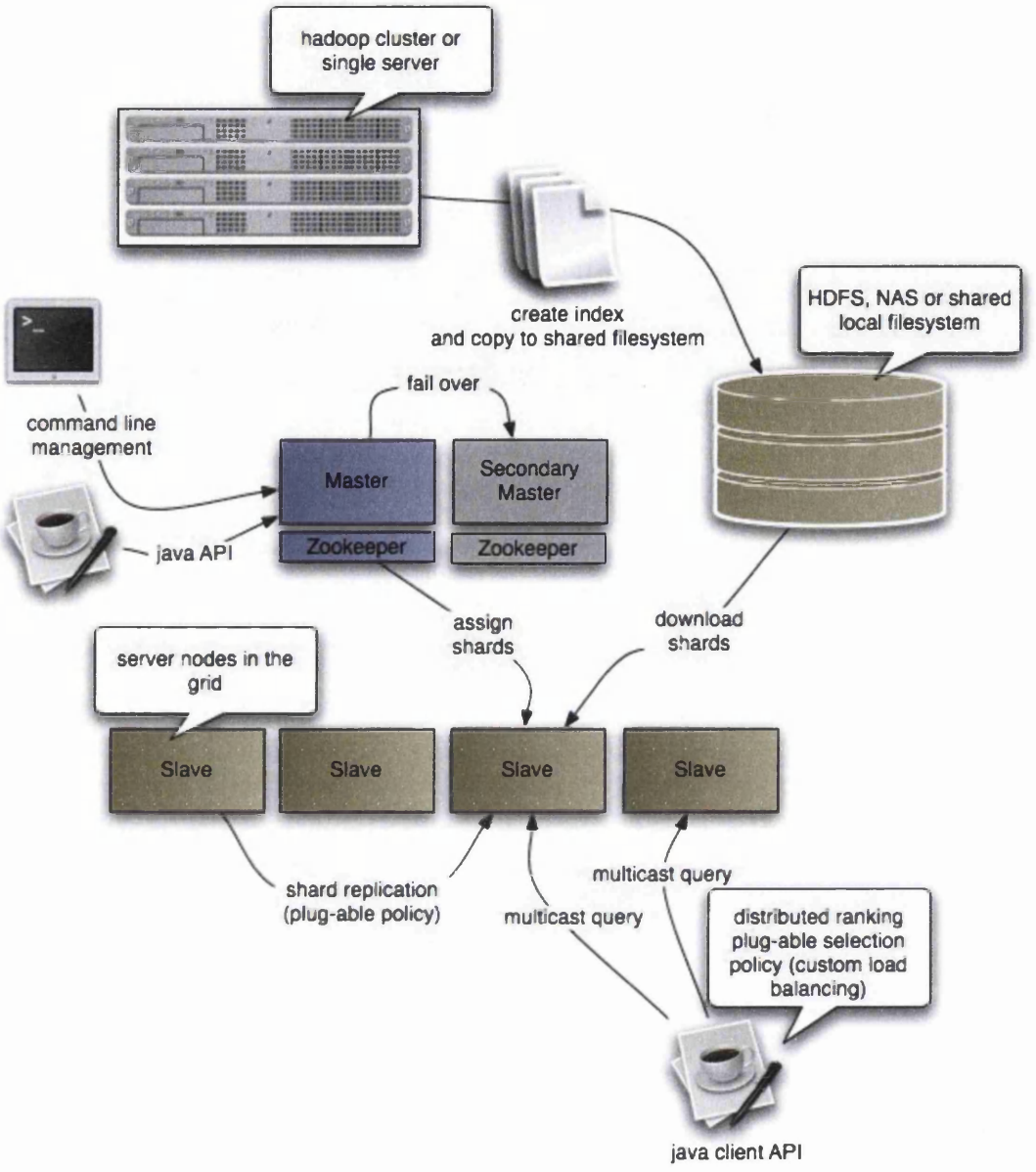


Figure 4.1 Showing how Katta integrates with Hadoop.

Whilst Katta offers some features which would be advantageous to the overall architecture the documentation is very lacking and weak. At time the current release version is 0.6 and has not had any code committed to the project for over a year.[1] So for this reason we decided to look for similar projects that were more suitable to our needs.

[1] https://github.com/sgroschupf/katta

4.4.7 Elastic Search

Elasticsearch is an open source search service based on the Apache Lucene project. It allows for a distributed architecture and near real-time retrieval.
The distributed nature and near real-time retrieval of elastic search really appealed to us. Whilst Pig offered a very flexible framework, its retrieval time was not suitable for our system. Lucene offers a great retrieval framework, though due to the amount of data being put into the system, we need a system which would scale horizontally. This is where projects such as Katta and Elasticsearch excelled.

Like Pig and Lucene Elastic search is accompanied by lots of documentation. This was one of the biggest advantages of Elasticsearch over Katta. Whilst Katta offered a good base to build a system on top of, its documentation was lacking depth and quality, as well it being a relatively new project, we were unsure about how stable the system would be, and if we were to encounter problems how we could fix them or how long it would take to fix.

One of the biggest advantages of elastic search is how it stores data. Elasticsearch allows the index to be either stored in-memory (no persistence) or on-disk(this is the default setting). Indexes that are stored in memory offer better performance. Obviously this is limited by the amount of physical memory available across the cluster.

As well as being scalable, fast, having good documentation and continual updates. Elastic search allows for the customization that we wish to implement. It allows us to do these customizations by passing parameters via JSON to it's HTTP API.

Custom Tokenizers
Due to twitter having a unique language, which special tokens, we needed a system that would allow us to use tokeniser that allowed for special tokens to be preserved. Due to Elasticsearch being built on top of Lucene this feature was available. (Elastisearch, 2014) This means rather than the following sentence

"Hi @jonhurlock, how are you? #question"

Being tokenized as:

['Hi','jonhurlock','how','are','you','question']

the special tokens (i.e. hashtags and mentions) are preserved so it would instead be tokenized as:

['Hi','@jonhurlock','how','are','you','#question']

This is an important feature as we know that users search using @mentions and hash tags. (Hurlock & Wilson, Searching Twitter: Separating the Tweet from the Chaff, 2011) (Teevan, Ramage, & Ringel Morris)

Custom Weightings

One of the requirements for the system is to allocate custom weightings to search results. For instance if a query returns a tweet which contains a link, we want to be able to say, because this tweets contains a link in is x times important that a tweet which is exactly the same but does not contain a link. This will happen at query time, to we need a way to be able to assign dynamic weightings to results.

Elastic search allows us to do this via the function Score Query, this allows the document to be boosted by a value in any field of a document / enriched tweet. The default behavior is to multiple the score by the desired score. However, it also allows other options such as replacing the score entirely, summing multiple scores, averaging scores, taking the max or min value, as well as having custom scoring scripts calculate a desired score. (Elasticsearch, 2014)

Due to all the advantages of Elasticsearch, we ended up choosing this for our implementation. In the next section, we discuss key components of the architecture in more detail.

4.5 Overall Architecture

In this section we discuss the final choices of architecture, and how we have setup the system.

Our system had to be designed to be robust, scalable, but also simple enough to allow us to write code that could be easily deployed and tested.

The system was to be run on commodity hardware, so had to be able to cope if one machine was to fail, either through network problems, or mechanical/electrical failure such as hard disk failure. The architecture and solutions we have chosen has allowed us to take into account these sorts of problems. Hadoop and elastic search whilst having certain nodes allocated to specific tasks, also allow for 'backup' nodes to gracefully take over if one node fails.

The system also had to be scalable allowing for expansion depending on the amount of data that is to be processed by the system. Hadoop and Elastic Search both scale linearly allowing more nodes to be added to cluster if more processing power is needed.

Both Elastic Search and Hadoop are predominately written in Java, and allow for customization. By having a system built with a language which is easy to learn, it allows easy access for us to customize it. One such customization we made was to do with the tokenization of strings within elastic search, allowing the twitter specific language to kept intact. One of the main advantages of hadoop was that it allows for programs to be written in other languages such as Python, Perl, Ruby and C++ (Noll, 2007) and then executed via the Hadoop Streaming API (The Apache Software Foundation, 2013).

The system ran on 10 machines. Three of which were bought through the project, whilst the remainder were old server blades sitting unused on the departmental network. We then repurposed these machines to be hadoop slaves.

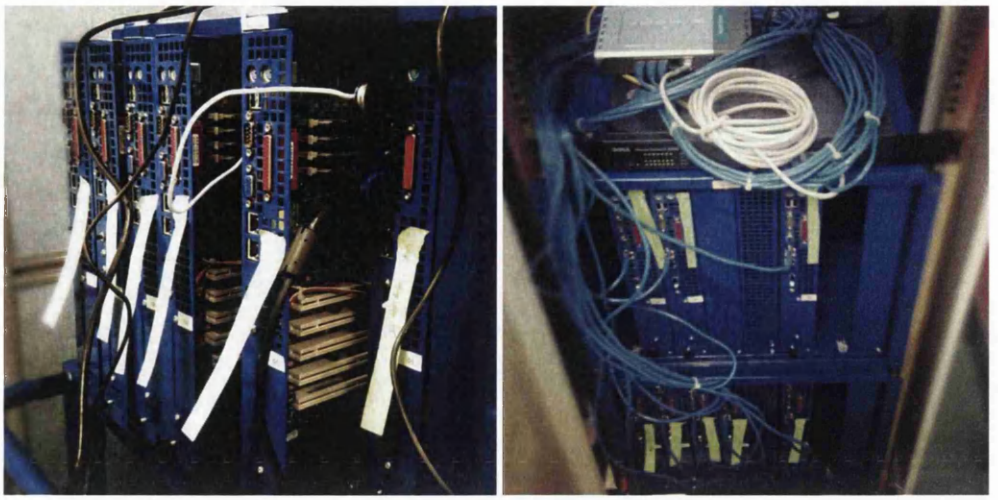


Figure 2.2 Photos of some of the Hadoop Cluster we built and ran during the Ph.D.

This took a considerable amount of time and effort, first checking each blade to see that all the physical components were actually working, then removing all data from the existing hard drives, installing a new operating system (ubuntu server), configuring the machine to it connected the university network, then installing hadoop on each machine, and configuring permissions for each machine to enable them to communicate with each other. Each machine was individually labeled so that we knew what each machine's role was. Over the course of the Ph.D. some of the machines experienced physical faults (hard drive failure, PSU failure). However, due to hadoop and elastic search's robustness and fault tolerance, the machines were constantly replicating data, and when one machine died, the other machine would share the load of the machine which had died. This is the case in all machines except for the name node, when this experience trouble. We had to restore from the Secondary NameNode.

In the next few paragraphs we will describe the main components of the architecture, splitting the system into two parts. Firstly the preprocessing part (hadoop work) and then the retrieval side of the system (elastic search).

4.5.1 Hadoop Architecture

In the diagram below we can see the physical architecture of the system, and how Hadoop was distributed over the cluster.

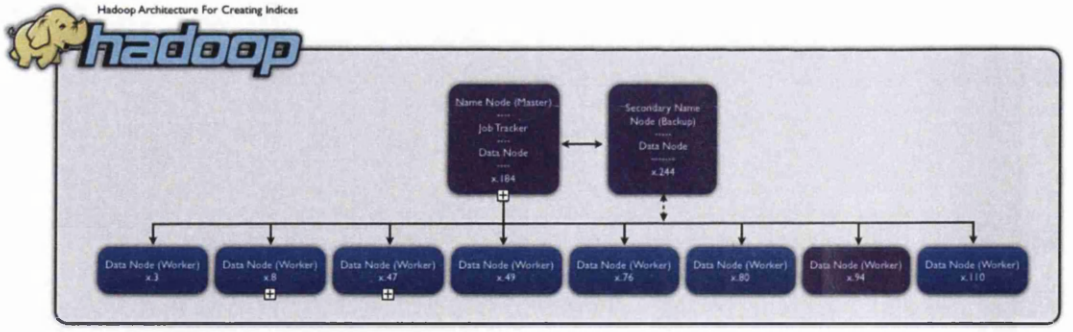


Figure 4.3 The Hadoop architecture we ran during this thesis.

Each machine is given a color that indicates what type of machine it is. Blue indicates it being on the departmental blades, whilst purple indicates a machine bough on the project.

Each node is also labeled with its role(s) as well as its IP address on the network. There several roles or daemons that run across a Hadoop cluster. These are:

- NameNode
- Secondary NameNode
- DataNode
- JobTracker
- TaskTracker

We will explain each of these roles. Hadoop allows for each machine to run single or multiple daemons. This allows for hadoop to be run over one machine in standalone, pseudo-distributed or full-distributed cluster.

NameNode

This NameNode acts as the brain of the cluster. Hadoop works on a master/slave architecture for both storage and computation. The NameNode is a master in this architecture.

It controls the filesystem (Hadoop File System / HDFS), providing tracking of how files are broken down into file blocks, where these blocks are stored across the cluster and monitors and maintains the health of the system.

Secondary NameNode

This is an assistant for monitoring the state of the cluster. Unlike the NameNode the Secondary NameNode in that it does not process real-time changes to HDFS. Instead it takes snapshots from the NameNode, allowing for a role back if the NameNode fails.

DataNode

As mentioned before Hadoop runs on a master/slave architecture. The DataNodes are the slaves in this architecture. The DataNode perform reading and writing of data to HDFS to actual files on the local file system.

JobTracker

The job tracker is the liason between the application running and Hadoop. Once a user has submitted a code to the cluster, the JobTracker determines the execution plan by determining which files to process, assigns nodes to different tasks and monitors all tasks as they're running. Should a task fail, the JobTracker will automatically relaunch the task, possibly on a different node, up to predefined limit of tries.

Task Tracker

As with the storage daemons, the computing daemons also follow a master/slave architecture. The JobTracker is the master overseeing the overall execution of a MapReduce job and the TaskTrackers manage the execution of individual tasks on each slave node.

Each TaskTracker is responsible for executing the individual tasks that the JobTracker assigns. Although there is a single TaskTracker per slave node, each TaskTracker can spawn multiple JVMs to handle many map or reduce tasks in parallel.

One responsibility of the TaskTracker is to constantly communicate with the JobTracker. If the JobTracker fails to receive a 'heartbeat' from a TaskTracker within

a specified amount of time, it will assume the TaskTracker has crashed and will resubmit the corresponding task to other nodes in the cluster.

4.5.2 Elasticsearch Architecture

As well as running hadoop the system also ran an elasticsearch cluster. Elastic search is a distributed variant of Lucene. Lucene is an Apache project, which provides high-performance, full-feature text search. Elastic search is a peer to peer based system, nodes communicate with each other directly.

The elasticsearch cluster was run across the same machines that the hadoop cluster was run on. There are three roles a node can play in the elastic search architecture. These three roles can be seen in the diagram below (load balancer, master node, work horses).

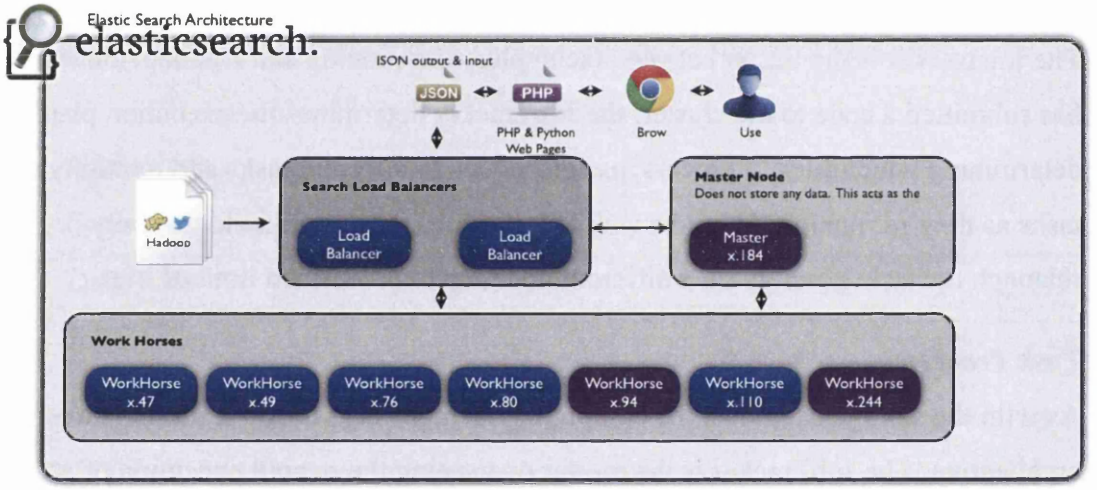


Figure 4.4 The Elastic Search architecture we ran as well as interaction points with users of the search engine.

Load Balancers

These nodes are used for distributing the workload across the cluster. They aim to optimize resource use, maximize throughput, minimize response time and avoid any overload of one particular resource. All queries are directed to one of the load balancers, which then assess which work horse(node) to run the query on. When we are adding data to the system, they are inserted via the load balancers, to make sure we do not overload any particular machine.

Master Node

All the main APIs (index, delete, search) do not communicate with the master node. The responsibility of the master node is to maintain the global cluster state, and act if nodes join or leave the cluster by reassigning shards. Each time a cluster state is changed, the state is made known to the other nodes in the cluster (the manner depends on the actual discovery implementation).[1]

Work Horses

Whist there is no name associated to nodes which are not load balancers or the master node, we have chosen to call them the work horses. These nodes deal with the requests put forth by the load balancers, and those made by the master node into relation of the status of the cluster.

4.6 Data Flow

Now that we have seen the main two components of the system, we will explain how data flows through the system. From Twitter to user retrieval.

There are six main steps in our system, these are listed below.

1. Obtaining Tweets from Twitter/Source
2. Classifying Tweets
3. Output Tweets regardless of classification (useful or not)
4. Indexing Tweets in ES
5. User Querying Index
6. Returning Ranked Tweets to User

At step 1, tweets are either gathered by one of our scripts. We have a script which scrapes the spritzer hose, as well as script which takes in a query as an input, then scrapes twitter Search API for tweets. We have tested the system with both, we have used the second method to help us calculate optimum weightings for each of our codes found in our initial study. We will talk about this in more detail in the following chapters. Both of these scripts generate flat files of JSON data.

[1] http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/modules-discovery.html

Step 2 now occurs, this is where the tweet JSON files get inserted into the Hadoop cluster, at this point all tweets in the files are classified based on the findings of our ICWSM paper. Towards the end of this step hadoop stores the output in HDFS.

Step 3 then occurs. We extract the output from step 2, removing the output from HDFS. The output consists of flat files, which contain JSON objects, each JSON object is the original tweet inserted into the system, with more meta data included to do with whether or not the tweet contains the codes identified in chapter 3.

Step 4, at this point we have flat files which contain the data we wish to retrieve data from. We need to index this data into elastic search. At this point data a script is run which indexes the JSON data into elastic search ready for retrieval.

Step 5. Now that data is indexed it is ready to be retrieved by a user or process. A HTTP request is made to elastic search from a client. At which point the request will hit one of the load balancers, the load balancer will then choose an appropriate node to fire the request to, this node will then retrieve the result. At which point Step 6 is initiated and a HTTP response is given back to the client.

It is worth noting we have made a search interface to allow keyword search to be performed by users to showcase the system. Screen shots can be seen in the figures below.

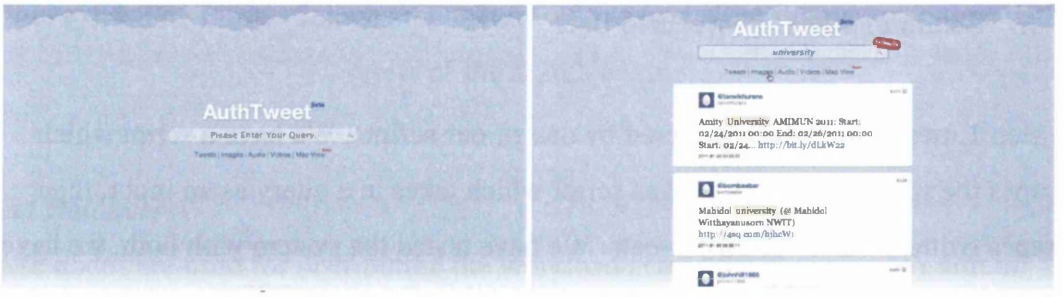


Figure 4.5 Showing screenshots of the search system in use.

4.7 Summary

In this chapter we have discussed, the architecture of the system, and rationale behind the choices we made to select this architecture. In the next chapter we will discuss how we determine if a tweet is either to be deemed useful or not, as well as how we rank each tweet. Specifically looking at the weightings for each of the codes.

Chapter 5: Automatically Identifying Usefulness

5.1 Introduction

In the previous chapters we have discussed a study which identified reasons for tweets being labeled as useful or not useful to people performing three types of common search tasks found on Twitter data. (See Chapter 3) We then introduced a distributed architecture in Chapter 4 that has allowed us to build a system that would allow us to index and search over twitter data.

In this chapter we describe the way in which we programmatically identified the codes described in Chapter 3. We borrow techniques from other research to enable to us perform some of the code checks we have implemented.

It should be of note to the reader that the architecture described in Chapter 4, is being utilized in this chapter to help process tweets. We modify tweet's JSON objects to include additional meta data about the tweet based on the codes described in Chapter 3. This is done via our Hadoop cluster described in the pervious chapter, to which the output is then fed into our Elasticsearch cluster after having custom weights applied to the JSON object, which are described in Chapter 7.

We utilize the architecture in Chapter 4, not only to speed up some calculations, but also to optimize for when collecting data from external websites. This is especially relevant when checking if a link is dead or not, and also to see what content is contained with a link. As it creates multiple (distributed) network request queues rather than have all requests being sent from one machine and having one large network request queue.

Due to codes being subject, query dependent and certain indexing techniques some of the codes described in Chapter 3 either not be detected or indexed. We give a brief overview of how well each of these codes has been implemented in the table below. We then go on to describe how we used techniques found in literature and techniques we identified to help automatically identify these codes.

Code	Level of Implementation
Useful Codes	
Experience	Fully implemented
Direct Recommendation	Fully implemented
Social Knowldege	Unable to implement at indexing time *
Specific Information	Implemented through various means
Entertaining	Not Implemented due to subjectiveness
Shared Sentiment	Partially implemented due to indexing time *
Time	Implemented *
Location	implemented *
Trusted Author	Implemented via SPAM detection
Trusted Avatar	Not implemented due to subjectiveness
Trusted Link	Fully Implemented
Actionable Link	Fully Implemented
Media Link	Fully Implemented
Useful Link	Implemented based on Lexical Quality
Retweeted Lots	Implemented *
Conversation	Implemented *
Not Useful Codes	
No Information	Partially implemented
Introspective	Partially implemented
Off Topic	Not implemented *
Too Technical	Not implemented *
Poorly Constructed	Fully implemented
SPAM	Fully implemnted
Wrong Language	Partially implemented *
Dead Link	Fully implemented
Time	Implemented *
Location	Implemented *
Untrusted Author	Fully implemented as inverse of trusted author
Untrusted- link	Fully implemented as inverse of tusted link
Perspective Oriented	Implemented
Disagree with Tweet	Not implemented due to subjectiveness *
Not Funny	Not implemented due to subjectiveness *
QnA	Fully implemented *
Repeated	Not implemented *

Table 5.1 Showing Codes from Chapter 3 that have been implemented.

* indicates that due to the stream, we cannot index this feature at index time, and can only be performed at query time, because it is either query or user specific.

5.2 Detecting Experience

Tweets that were labeled as being useful contained personal experience. We used natural language processing techniques to look at the structure of tweets which

contained personal experience, and identified two ways in which personal experience was being conveyed.

The first way was when a tweet contained a pronoun (I/We), followed by 0 or 1 verbs (e.g. have) or a contraction of these to produce something such as I've or We've. This would then be followed by a verb which was either in the past of presence tense.

The second way we identified tweets as containing personal experience was when a tweet contained a sentiment rich word (verb or adjective such as hate/hated, bore/bored, love/loved) followed by a pronoun describing themselves or another object e.g. my/myself. A list of sentiment rich words was curated by ourselves, which contained present and past tense words.

The above detection was performed using regular expressions, and resulted in a binary score, of either containing an experience or not containing an experience.

5.3 Direct Recommendations

Another of our codes found in useful tweets was direct recommendation. We again look at the sentence structure of tweets to estimate if the tweet contains direct recommendation.

In this instance a person must be identified, either by using their twitter username e.g. @jonhurlock or using a pronoun word such a 'you' or 'we', followed by a modal verb with either present or future tense.

This again was detected using a regular expression, giving a binary score to either a tweet containing or not containing a direct recommendation, with 0 representing no direct recommendation and 1 representing that the tweet contained a direct recommendation.

A list of commonly used pronouns and modal verbs was manually created by the author to facilitate this detection. As the POS tagger we were using (NLTK) did not allow for identification of modal verbs.

5.4 Social Knowledge

We defined tweets with Social Knowledge as containing information that is spreading socially, or becoming general knowledge. This code is temporally relevant, as it is query time specific, due to this fact we were unable to process it utilizing the Hadoop cluster we had setup.

5.5 Specific Information

The second most frequent of codes encountered in Chapter 3 was that of specific information. We had several tests to see if a tweet contained specific information. We defined what specific was based on the following factors time, price, and mention of a proper noun.

We detected if a time was mentioned by using regular expressions to detect combinations of time and dates, as well as detecting words based on a dictionary of words to do with mentions of time and dates. Examples of the some of the terms we detected are mentioned in the following paragraphs.

Day time descriptors are terms describing points within a day. Examples are:
morning|mid-day|midday|afternoon|evening|night|dusk|dawn

We searched for descriptors describing times within the week:
today|yesterday|tomorrow|weekend|week|midweek
monday|tuesday|wednesday|thursday|friday|saturday|sunday|mondays|tuesdays|wedne
sdays|fridays|saturdays|sundays||mon|tue|tues|wed|weds|thu|thurs|fri

Words to do with the month:
january|march|april|june|july|august|september|october|november|December|month|mo
nthly|jan|feb|mar|apr|jun|jul|aug|sept|oct|nov|dec

When implementing the code initially we were over reporting time matches, this was due to the term "may" appearing. Although in our test cases we did not find any tweets containing the term "sun" or "sat" we believe we potentially could have over reported on this too. So when implementing this, mentions of the term "sat", "sun" had to be prefix by the terms this|next|last or alternately then have another mention of time or date in the tweet, for the tweet to be labeled as containing time based information in it. If the term "may" had a numerical date, and an optional ordinal suffix (e.g. nd, th, st) preceding it, we deemed this to be a mention of a date.

As well as the fore mentioned detection, we also built two regular expressions to capture whether a user mentioned a time. These are given below:

General time mention:

(([|.|,|-|_||(|)|!|*])([0-2]?)([0-9])( )?(am|pm|a.m|p.m|a.m.|p.m.)([|.|,|-|_||(|)|!|*]))

24 Hour clock detection:

([|.|,|-|_||(|)|!|*])([0-2]?)([0-9]):(([0-5][0-9]))([ ]?)([am|pm|a.m|p.m|a.m.|p.m.]?)([|.|,|-|_||(|)|!|*])

Detection of prices was done again via regular expressions. We looked for patterns where by a currency symbol (e.g. £) or code (GBP) was either proceeding or after a number.

As well as mentioning times and currencies we also looked for mention of proper nouns. A proper noun is a word that refers to a unique entity, for instance a name of a city, a person's name or a company name. We count @mentions as proper nouns, however if the 'in_reply_to_status_id' meta field is included we ignore any mentions, this is due to the person via a @mention is not being referenced and is instead being used as a linking mechanism.

We detected proper nouns by utilizing NLTK's POS tagger, this POS tagger is trained on the Brown Corpus which contains 500 samples of English-language text, compiled

from works published in the United States in 1961. (Bird, Klein, & Loper, 2009) (Wikipedia, 2015)

At the time of writing the code detection we looked to see if anyone had written a POS tagger aimed specifically for Twitter data. Unfortunately at the time we could not find such a tagger, though since completing the implementation, several taggers have been created with varying levels of success. (Owoputi, O'Connor, Dyer, Gimpel, Schneider, & Smith, 2013) (Derczynski, Ritter, Clark, & Bontcheva, 2013)

Twitter poses interesting questions to how we tag, certain terms, e.g. emoticons, emoji as well as the question of how we tag terms that are 'new' or are intentionally misspelt or elongated. E.g. 'nooooooo' or 'yeeeeesssss', which are misspelt and elongated to provide emphasis.

We performed some preprocessing of the text, to remove hashtags from hashtags in tweets, for instance '#something' just became 'something'. As we believed that some valuable information maybe hidden within hashtags. We also converted @mentions to real names, such that a tweet which look like '@user1 and @user2 liked @user1's car' became 'Alice and Bob liked Alice's car'. We did this to try and naturalize the sentence to something that would appear in the Brown corpus.

Whilst the Brown corpus is outdated, it would be able to detection proper nouns in terms of location and people's names, but would not necessarily be able to decipher whether the term 'Apple' was proper noun (Apple the company) or a common noun (the fruit). As well as date, the corpus is based on high quality texts, whilst we know the quality of text with in the twitter-sphere may not be as high quality. (Rello & Baeza-Yates, 2012)

Utilising the methods mentioned in this subsection we were able to extract three binary scores one for proper nouns (individual's names, places and/or organizations), one for price and one for time.

5.6 Entertaining Tweets

We are unable to classify tweets as entertaining due to the subjective nature.

5.7 Shared Sentiment

Whilst we can't detect if sentiment matches the personal sentiment of the user performing whilst indexing the data, we can detect whether sentiment is being displayed in the tweet.

We detect sentiment utilizing two different machine-learning approaches. We use a naïve Bayesian classifier and a Fisher's classifier.

Naïve Bayesian Classification is a machine learning technique that calculates the probability of a document belonging within a 'bucket' (classification). It is called naïve because the algorithm assumes that the probabilities being combined are independent of each other. (Segaran, 2007)

Fisher Classifier, named after R.A.Fisher, is a probabilistic approach to classification. Unlike the naïve Bayesian classifier approach, whereby feature probabilities are used to create a single classification for the document. The Fisher method calculates the probability of a classification for each feature in the document, then combines the probabilities and tests to see if the set of probabilities is more or less likely than a random set. (Segaran, 2007) We intern get a probability score for each bucket/classification for each document. We took the bucket with the highest probability as the sentiment for a given tweet. If probabilities between a strong sentiment (either positive or negative), were equal to either (irrelevant or neutral), we took the stronger sentiment. However, if the probability of the strong sentiments were the same, we decided that the sentiment was neutral, as they cancelled each other out.

We trained both the classifiers on an existing set of 5513 hand labeled data from Sander Analytics. (Sanders Analytics, 2011). Tweets were either classified as Positive, Negative, Neutral or Irrelevant. However, due to the way in which this corpus was generated, it has some bias. Not only in terms of size, but also content. The Sanders corpus, was generated from search results pages which searched for the following terms "#Google", "#Microsoft", "#Twitter" and "@Apple". Ideally we would have liked to have had a large sample of tweets which had no term bias, as well as having these tweets manually labeled by expert judges. However, due to the lack of

time and budget, we decided to utilize this existing resource. At time of implementing this was one of the largest manually labeled twitter datasets available in terms of sentiment.

When training the naïve Bayesian classifier, we used a unigram approach, taking each term as a feature. This assigns each trained term a weighting/probability for belonging to each of the sentiment buckets.

We performed some preprocessing before submitting the data to our classifiers, as well as when we trained our classifiers. We replaced all @mentions and links with the terms person and link. We deemed @mentions and links to be free of sentiment, and did not want the classifier to 'learn' certain @mentions or links as having sentiment. Hashtags also get their '#' removed e.g. #love becomes love.

We also manipulate all twitter entities and words that have consecutive repeating letters. We reduce the number of consecutive repeating letters so there is a maximum of two repeating consecutive letters per word e.g. helllllllllo would become hello. We did this in an attempt to minimize the amount of terms that haven't been seen by the classifier.

When recording the output of the sentiment analysis, results were recorded as positive being 1, negative being recorded as -1, and both irrelevant and neutral being recorded as 0. We decided to combine irrelevant and neutral for ease of coding, and allowing us to assign a scalar value to the code. We recorded sentiment scores for both types of classifier.

5.8 Time

As mentioned in the specific information section, we are able to detect when a user is talking about a point in time, by using regular expressions at index time. However, we wanted to know if the user is talking about a current event if the tweet is deemed to be useful, or if the time is said to 'outdated' if the tweet is deemed to be not useful. These are both subjective, and query time dependent variables, thus it we are not able to be index this at index time.

To counter this we have decided to split any mentions of times into three categories past, present and future.

All tweets that contain a time, which could be considered up to an hour before posting, we count as present. We do this due to the user may not have had signal when the event occurred and are mentioning something that is 'nearly' current. Any mention of a time +6 hours from time of positing is also counted as present.

We encountered some false positives, such as if we see a tweet saying 'Party estimated to finish 2:00AM' posted at 22:00, we can read that the event is running from 22:00 on day 1, to 02:00 on day 2. So will not be detected as present or future it will instead interpret this as the past. It does this, because there is no mention of 'tomorrow' or a reference to day+1. We calculate day+1 and day-1, based on the tweets creation date, and any mention of the day either side of it. E.g. if a tweet is posted on Monday the 14th of August, if it mentions Sunday (with a past tense verb), Tuesday, 13th (with a past tense verb) or 15th, we infer we are talking about an event within a 24 hour period surrounding the current date, we then try to estimate whether the date and time is within the 6 hour time frame if it is within the 24 window, or alternatively an hour if it is in the past.

If no mention of time is mentioned no value is entered, if it is in the future, we give it a value of 1. If it is in the past we give it a value of -1, and if it is in the present we give it a value of 0.

5.9 Location

Like the time code, location is again query dependent. Entity disambiguation is a big problem. If I ask someone where London is. It is logical that someone in Europe would say London is located in the UK. However if I ask someone in Canada, they may say Ontario, very few clues are given at query time as to where a person is actually trying to refer to. One of the biggest clues may be using the accept-language header in the user's HTTP request. However there is a disproportionate amount set to en-us, and in the case of London, this does not help us if I mean London, UK.

Instead of trying to tackle the problem of location disambiguation, and matching between tweet and query, we instead search for proper nouns in tweets, and of those, we see if any of them are locations via the google geocoding API. Though we are now rate limited to 2,500 request per hour. Thus like the Link analysis described later, we have a 'caching' service to see reduce the number of requests sent.

Whilst we don't perform any further analysis on this code as it is query specific, we do record the longitude and latitude returned from the Google geocoding API based on the first longest noun-chained string, we pass it.

A noun-chained string is where we have a noun followed by a noun, followed by another noun and so on. We take this n-gram and pass it to the API, we also generate a list of sub n-grams for the original n-gram, until we are left with a list of unigrams, and pass all of these to check if there is a result.

We use the longest, first returning n-gram, we do this because Alice London England Bob is 4 proper nouns, however, London England is the n-gram we want.

This in theory could be used to attempt to produce a measure between query location and locations mentioned in the code, though this is query time specific, so is not implemented.

5.10 Trusted Author

Trusting an online entity is a very subjective matter, people place trust in certain attributes that others may disagree with. When we described trusted author in chapter 3, we described it as a 'twitter account has a reputation/following'. Whilst there have been papers such as Measuring Influence in Twitter: the million follower fallacy (Cha, Haddadi, Benevenuto, & Gummadi, 2010), saying that audience size doesn't prove that someone is more influential. We found that people still put a lot of emphasis on these numbers via the study described in Chapter 3.

As people decided to put their trust in these numbers, we have decided to utilize a reputation metric developed by Wang (Wang, 2010) , which we feel reflects the comments made by participants.

$$R(v_i) = \frac{d_I(v_i)}{d_I(v_i) + d_O(v_i)}$$

Which says reputation(R) of a user is equal to the number of indegrees (followers) divided by the indegree (followers)+outdegree(following) of users. We implemented Wang's scoring metric into codes, we were able to extract all these numbers from the JSON object when processing the tweet.

5.11 Trusted Avatar

Trusted based on avatars is also very subjective. Work has been carried out on understanding trust based on facial avatars, however, due to users can upload pictures of anything as long as it adheres to the terms of use, we find that it would be unfeasible for us to try and extract trust based on user avatars.

5.12 Detecting Questions in Tweets

Like other social network services, micro blogging platforms facilitate interaction among people. Boyd and Ellison note these interactions often entail reinforcements and maintenance of social ties that were created in more traditional venues. (Boyd & Ellison, 2007)

Morris et al, (Morris, Teevan, & Panovich, 2010) report a detailed survey of people with respect to question-asking and question-answering behavior on Twitter. Their analysis suggests that in many cases people turn to their Twitter network to help them resolve information needs. In these situations users rely on Twitter as an informal social search service.

Twitter is full of idosyncrasies, which makes processing it difficult. On the other it is very restricted in length and tends to employ simple syntactic constructios, which could help wthe performace of NLP processing.

Evans and Chi (Evans & Chi, 2008) analyze social search interactions under the lens of Broder's taxonomy of search (Broder, 2002):transactional, navigational and

informational. They note that social interactions entail an especially promising tool for searchers with informational needs. i.e. people trying to gather information, aspopposed to people trying to accomplish a particular task(transactional) or find (navigate to) a particular Web resource.

Of particular interest is the dissection of social search tactics outlined in Evans & Chi's work. (Evans & Chi, 2008) The authors make a distinction between targeted asking and public asking.

Targeted asking includes modalities such as email, where a searcher directs a question to a particular individual or delineated group. On the other hand, public asking involves broadcasting a question to a wide audience, either through posting a question to a wide audience, either through posting a question to a public feed on Twitter, or by enlisting a search service such as Aardvark (vark.com). Targeted asking in Twitter can be accomplished by the use of the @ symbol or through the use of direct messages (DMs), which created a private conversation between the sender and receiver.

When people ask questions on twitter they typically do so in a fashion that lies somewhere between targeted and public asking. Excluding direct messages, which are private, and will not be covered in this thesis, questions on Twitter are posted to all of a user's followers, and therefore have a significant public component. On the other hand, questions are only available to a user's self-selected followers, thus limiting the scope of the question audience. Directing a question to a particular follower via an @ mention signals the user's intent that his or her question has a narrow target, but it presence on the public feed (Rather than a private direct message) means that the question is serving another purpose within the ongoing exchange between user and followers.

A causal perusal of Twitter shows that people use the service for many reasons, including social search.

Not all "questions" on Twitter end with a question mark. Indeed the linguistic literature on the semantic of questions is large. Here we enlist a portion of that literature to help us operationalize the idea of a question in order to draw a meaningful sample of user questions form our corpus of Twitter data.

To guide our analysis, we refer to Karttunens (Karttunen, 1977) description of question embedding verbs in Karttunen 1977. Question embedding verbs are phrases that lend a declarative sentence interrogative semantics. The sentence I would like to know where you will be after the plenary is, in Karttunen's analysis the same as asking, Where will you be after the plenary?

To the best of our knowledge, no canonical list of question embedding verbs exists. Thus we combined an analysis of the verbs listed by Karttunen and our own reading of a large number of tweets to arrive at the following working definition of what constitutes a question in our analysis. A tweet contains a question if:

- It contains a question mark that is not part of a URL.
- It contains the phrase I* [try*,like,need] to find
- It contains the phrase I* [try*,like,need] to know
- It contains the phrase I*m looking for
- It contains the phrase I* wonder*

In these cases the * sign is a wildcard, signaling 0 or more instances of any character. The list above is admittedly ad hoc, but our initial analysis focuses on tweets that match these patterns yielding plausible samples.

After implementing this solution further work has been carried out to detect questions within bodies of text.

Wang and Chua used syntactic shallow pattern mining in an attempted to automatically detect questions in online content. Whilst Dent and Paul applied a different NLP approach in an attempt to automatically detect questions on twitter which an accuracy of .67881. Dent and Paul commented that their scores were rather

low and indicated that well formedness is probably not a good indicator of information seeking questions on Twitter.

We based our question detection algorithm on the work carried out by Efron and Winget (Efron & Winget, 2010). We were able to produce a program which cleaned the data, stripping out punctuation and other entites such as URLs, special symbols such as @ and #. After cleaning the data regular expressions based on the work of Efron and Karttunen are applied to see if we can detect questions.

Any twets which contained a question, were given a QnA score of 0.5 indicating we had found a question, if a subsequent tweet had the meta data with the 'in_reply_to_user_id' field as that tweet's id, we would then give both tweets a score of 1.

5.13 Conversation

One of the codes we identified was the conversation code. The code says that the tweet is part of a series of tweets, and they all need to be useful. Whilst we can not at index time decide if all tweets are considered useful, we can identify tweets which are in reply to another tweet utilizing the 'in_reply_to_user_id' meta data. If the tweet contains this then we automatically mark this tweet as part of a conversation, also if another tweet references a pre indexed tweet (via searching for it's tweet id), we mark both as conversation (via a HTTP PUT statement in elastic search).
We recorded this as a binary measure 1, being, part of conversation, 0 being not part of a conversation.

5.14 Link Analysis

A lot of the codes to do with a tweet being either labeled useful or not useful were to do with the links that were either prevalent or missing from a tweet. We even dedicated a category to links in our Useful codes, and identified Dead links as a reason for a tweet being deemed not useful.

At the time of coding we used regular expressions to enable us to extract links from tweets. The process has now become simplified with twitter automatically extracting links and now including them in the list of entitles found in a tweet's meta-data.[1]

5.15 Performing Link Analysis via HTTP Response Headers

We perform many types of checks on links to see if a tweet is to be deemed useful or not. In this section we describe how we utilize HTTP response codes, to classify links.

Below is a sample of HTTP response headers[2]. Responses are grouped into five different classes: information responses, successful responses, redirection, client errors and server errors.

---

[1] https://dev.twitter.com/overview/api/entities-in-twitter-objects#urls

[2] https://developer.mozilla.org/en-US/docs/Web/HTTP/Response_codes

Status Code	Status Text	Description
Informational Responses		
100	Continue	This interim response indicates that everything so far is OK and that the client should continue with the request or ignore it if it is already finished.
101	Switiching Protocol	This code is sent in response to an Upgrade: request header by the client, and indicates that the protocol the server is switching too. It was introduced to allow migration to an incompatible protocol version, and is not in common use.
Successful Responses		
200	OK	The request has succeeded. The meaning of a success varies depending on the HTTP method: GET: The resource has been fetched and is transmitted in the message body. HEAD: The entity headers are in the message body. POST: The resource describing the result of the action is transmitted in the message body. TRACE: The message body contains the request message as received by the server
201	Created	The request has succeeded and a new resource has been created as a result of it. This is typically the response sent after a PUT request.
202	Accepted	The request has been received but not yet acted upon. It is non-committal, meaning that there is no way in HTTP to later send an asynchronous response indicating the outcome of processing the request. It is intended for cases where another process or server handles the request, or for batch processing.
Redirection Messages		
300	Multiple Choice	The request has more than one possible responses. User-agent or user should choose one of them. There is no standardized way to choose one of the responses.
301	Moved Permanently	This response code means that URI of requested resource has been changed. Probably, new URI would be given in the response.
302	Found	This response code means that URI of requested resource has been changed temporarily. New changes in the URI might be made in the future. Therefore, this same URI should be used by the client in future requests.
Client Error Responses		
400	Bad Request	This response means that server could not understand the request due to invalid syntax.

401	Unauthorized	Authentication is needed to get requested response. This is similar to 403, but in this case, authentication is possible.
403	Forbidden	Client does not have access rights to the content so server is rejecting to give proper response.
Server Error Responses		
500	Internal Server Error	The server has encountered a situation it doesn't know how to handle.
501	Not Implemented	The request method is not supported by the server and cannot be handled. The only methods that servers are required to support (and therefore that must not return this code) are GET and HEAD.
502	Bad Gateway	This error response means that the server, while working as a gateway to get a response needed to handle the request, got an invalid response.

Table 5.2 Showing HTTP Response Codes

Checking URLs is possibly the biggest bottleneck in our system. As the time it takes to check an external resource takes along time. We have tried to minimize this bottleneck by introducing the following architecture.

We perform a HTTP Response header check on all previously unseen URLs that are contained in tweets. We do not perform multiple checks on URLs due to this would increase the time to perform processing of all tweets.

A list of URLs is saved into a MySQL database along with its response code, and all other relative data to do with indexing, so that when it comes to indexing, we can quickly see if a URL has been indexed before or not. One down side of this however is, if the URL becomes dead or alive, anytime after indexing, our corpus is out of date.

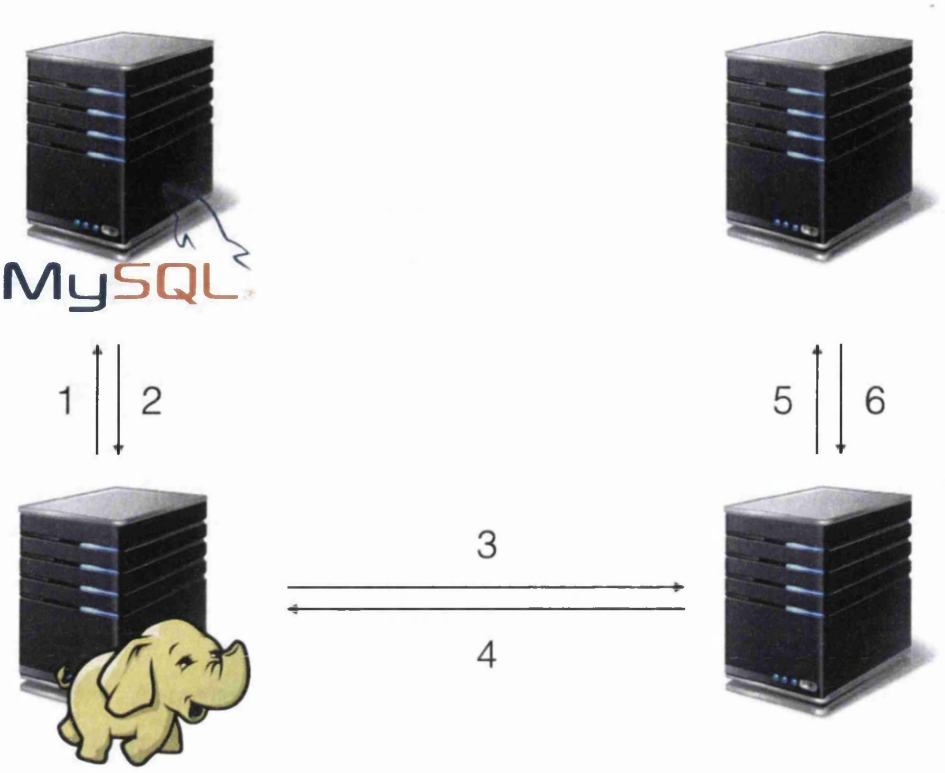


Figure 5.1 Showing how our system checks and caches links

The following processes occur during indexing. Step one involves the indexer builder located on the hadoop cluster, detecting a link. When this is triggered, a request is sent to our MySQL database (1) to check if the URL has been seen before.

Information then flows from the MySQL database back to the indexer (2), if the link has been seen before, then the indexer, indexes based on the data held in the MySQL database.

If the link has not been seen before then, the indexing program will attempt to index the URL. A request is sent to the URL from the hadoop cluster (3), it then waits for a response (4), if the response is in the 200 range (successful response range) then we perform a scrape of the content as long as it does not fall in the media link category, this is further described later in this chapter. The reason for the scrape is to see if the link falls into the actionable link or useful link category

The relevant data is collected and the tweet is marked appropriately, an entry for the link is then put into the MySQL database, to allow for this link to 'cached' and allow for quicker indexing.

If at stage 4 the HTTP response header is in the redirection range (300-399) then we attempt to follow the redirection of the URL (steps 5 and 6), we set a limit on upto three redirects to stop any link cyclic links (see image below) looping and freezing the indexing, as well as to reduce indexing time.

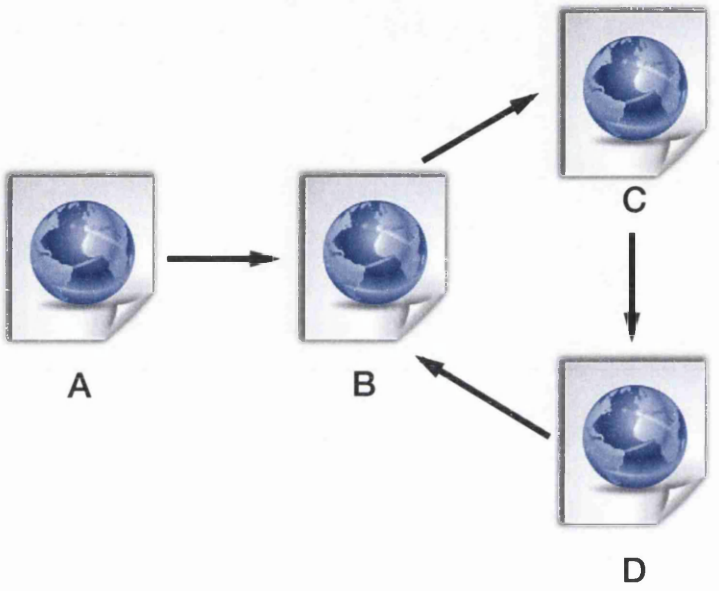


Figure 5.2 Showing cyclic redirection links

Above is an example of cyclic links, where page A redirects to B, B to C, C to D, and D to B. If we were to try and index content on A we followed all redirects, we would never be able to index this data as we would be continually chasing links. If the there are more than three redirects we declare the link to be dead.

During the redirection stage we also ping the MySQL database to see if the redirect links have been already indexed in an attempt to speed up indexing.

If a redirect does return a page in under three redirects of more, we then scrape the endpoint for indexing just as we would do if a page returned a successful status code,

and we add the redirection path to the MySQL database, to prevent us from having to perform the full redirection process again.

Below, we can see output from the program, showing the redirection process. It shows the extraction of the URL from the original text.. The URLs it found, Then any 'live'/successful responses it found, as well as any URLS which have been found via a redirection process, The redirection process it has been through, and any URLs which are deemed to be dead.

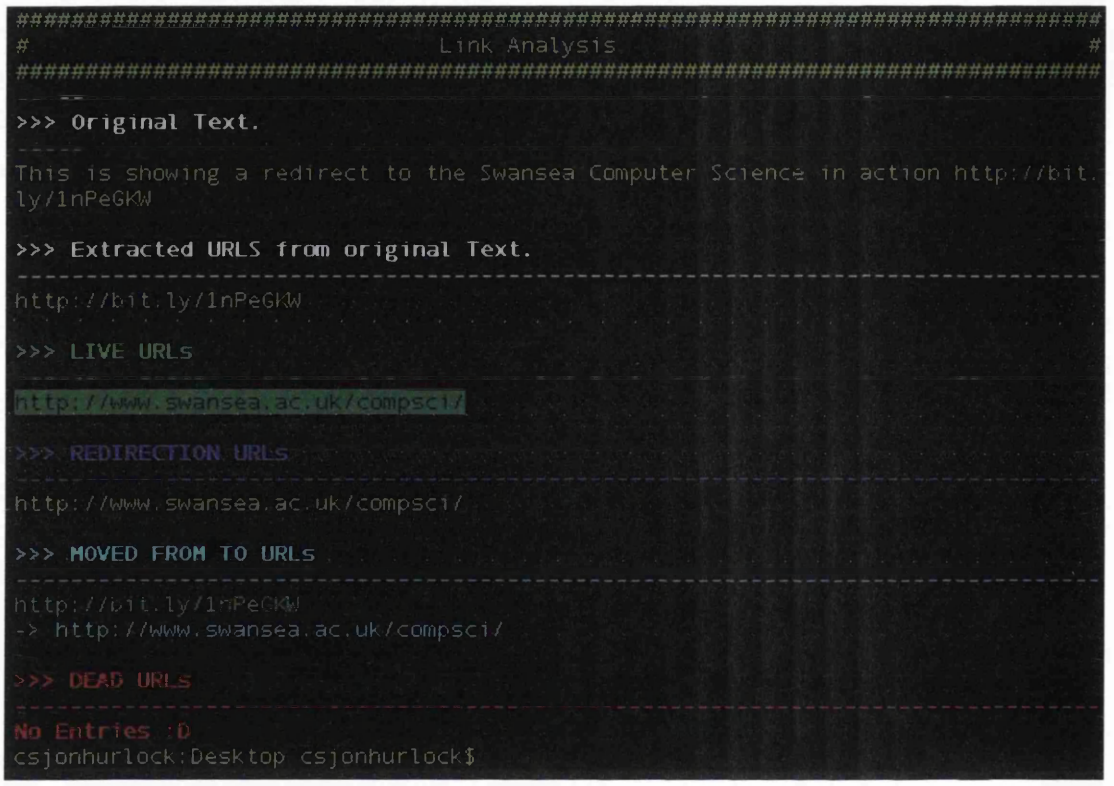


Figure 5.3 Showing our link analysis tool extracting and checking links

One of the most important checks that we perform is to see if the URL is dead or unreachable, this was highlighted in chapter 3, as one of the reasons a tweet would be deemed to be not useful. We classify any response in the 400 (Client Error) and 500 (Server Error) range to be deemed as a dead link.

We give an over all score for each tweet in terms of dead links, this is calculated as the sum number of dead links that are contained within the tweet.

5.16 Media Links

When it comes to classifying a link as media link, we use a list of regular expressions which checks for media types, as well as checking the URL against a list of known media services such as youtube, instagram, twitpic, BBC iplayer, as well as a number of other popular media serving sites. The original list was based on a list of photo sharing services listed by Wikipedia[1] as well as a list of video hosting services created on Wikipedia[2]. Some additional services, such as BBC iplayer were added.

If the link returns a successful response, and one of the regular expressions is matched we deem this to be a media link and no scraping of the content occurs. It its worth mentioning we detect images, movies and audio based media. When we record this code, it is stored as a binary value, if any links contain a media link, a value of 1 is stored, if no media links are found a value of 0 is stored.

5.17 Trusted Links

Trust is an important factor according to the study performed in chapter 3. We discovered that the trust a user assigns to a link can either make the tweet be useful or not useful. Looking through the literature there are many examples of systems which crawl the web and assign trust scores to webpages, however there is little literature on how people assign trust to links.

BJ Fogg (Fogg, et al., 2001) has produced several publications with peers looking at how people assign credibility to computing products and assess the credibility of web sites.

Prominence-Interpretation theory suggested by Fogg, posits that two things happen when people assess credibility online. Firstly the user notices something (prominence) and secondly they user makes a judgment about it (interpretation).
If one or the other does not happen, then there is no credibility assessment. The process of noticing a prominent element and making an interpretation happens more

[1] https://en.wikipedia.org/wiki/List_of_photo-sharing_websites

[2] https://en.wikipedia.org/wiki/List_of_video_hosting_services

than once when a person evaluates a web site, with new aspects of the site being noticed and interpreted as the user makes and overall assessment of credibility.

Fogg suggests that there are at least five factors that affect prominence.

1. Involvement – of the user (i.e. the motivation and ability to scrutinize Web site content)
2. Topic – of the Web site (e.g. news, entertainment)
3. Task – of the user (e.g. seeking information, seeking amusement, making a transaction)
4. Experience – of the user (e.g. novice vs. expert in regard to the subject matter or web conventions)
5. Individual Differences – (e.g. a person's need for cognition, learning style or literacy level)

Fogg also stipulates that there are various factors which influence Interpretation.

1. Assumptions – in a user's mind (i.e. culture, past experiences, heuristics and so on)
2. Skill/Knowledge – of a user (e.g. user's level of competency in the site's subject matter)
3. Context – (e.g. the user's environment, user expectations, situational norms ans so on)

Promience-Interpreation theory, is subjective, individualist and task oriented. We cannot program a subjective trust score, nor can we account for each user of our search system. In the paper entitled elements of computer credibility Fogg and Tseng, make reference to the following factors being cues for credibility – Familiarity and Social Status.

We have thus built a trust system that concentrates on the familiarity and social status of a link. The way in which we have programmed this, is to gather a list of the top 1000 domains being accessed by a country for a given time frame, then determining if

the link is in the list of top domains being accessed for a given time frame. The score is binary, either it is trusted or not trusted. With 1 being trusted.

The list of domains is gathered from a third party – Alexa Internet. Alexa is a subsidiary company of Amazon.com which provides commercial web traffic data. It collects this data via either the Alexa/Amazon toolbar or via javascript which is embedded into the webstie.

5.18 Actionable Links

An actionable link is defined whereby a user can perform a transaction by using the link. An example of this might be making a purchase, or filling in a form to complete a transaction. To define whether a link was actionable, we first had to find out if the link returned a successful HTTP response.

If we receive a 200 ranged HTTP response, then we attempt to scrape the content of the webpage. We then check to see if there are any form elements via counting form elements within the DOM. This is done via the beautiful soup library for python, allowing use to use CSS selectors to count the number of form elements.

If there are 0 form element the link is labeled as not actionable (0), if there are form elements, we then calculate the average number of inputs for each form on each form. This involves check for input, button, submit, textbox, checkbox, dropdown and radio elements within a form element (we do not include hidden form elements).

If this number is above a threshold of 3, we state that the page is probably actionable. Allowing for a transaction to be made. As for recording a score with give it a floating point value, which is calculated as the mean number of inputs per form.

5.19 Useful Links and Lexical Quality

We defined a useful link as a link which provides us with valuable information content e.g. authoritative, educated reviews and discussions. To detect that the link provided valuable information, we decided to run a lexical analysis on the content. We stipulated that links which provide valuable information will have a high lexical

score, and again information from authoritative sources will also have a high lexical score. The way in which we calculated the lexical score was based on work by Rello and Baez Yates (Rello & Baeza-Yates, 2012). Where lexical quality is defined as follows:

$$LQ = mean_{\omega_i \in W}\left(\frac{df \quad misspell \quad \omega_i}{df \quad correct \quad \omega_i}\right)$$

Where W is a set of frequently misspelled words. Those words were chosen such that they were frequent and had large relative error. They then use data from a leading search engine to estimate the document frequency (df) values, computing the relative ratio of the most popular misspells to the correct spellings, averaged over a word sample W.

We did not have a large document base from a leading search engine to estimate document frequency values so instead we have stayed within document, and have done an inverse function, so instead of the score 0 showing perfect lexical quality, 1 instead reflects perfect lexical quality. If a link was dead a score of 0 was given to this link as its usefulness score, as there was no data to be retrieved.

Originally we used a machine learning approach to see if we could detect misspells, with the training data being sourced from texts on project Gutenberg. We then changed our approach as when performing simple test the run time and results we tested gave poor results. Instead we utilized a mixture of NLTK and wordnet as well as the python enchant package which allowed us to check the spelling of words.

As we already have a check to see if a URL is misformed through the deadlink check, we also had to check lexical quality of tweets. We check lexical quality of tweets to detect poorly constructed tweets. The detection of this code was performed by utilizing the method described above.

5.20 Retweeted

Although we can not see if a link is retweeted at index time due to us performing batch processing (Hadoop), we can update the retweet_count meta data after

processing the data in Hadoop and add it into elastic search. This is done by just incrementing the retweet count value every time a tweet is retweeted. This will only work in a system which continually runs. We increment the retweet_counter, when we observe a tweet which has meta data stating that it is a retweet, and contains the retweeted tweet's original id. By default at index time this given a value of 0.

5.21 Summary

In this chapter we have identified how we programmatically extract the codes described in chapter 3 utilizing the architecture we described in chapter 4. We have not been able to automatically extract all codes due to the way in which we process data, the subjectivity of some of the codes as well as technical limitations (e.g. trusted avatar).

In the next section we introduce a dataset we have collected which will enable us to compare our system against others as well as allow us to optimize how we assign weightings (importance) to each of the codes.

Chapter 6: Building a Test Data Set

So far in this thesis we have discussed how we have built a system which is able to index and retrieve large data streams (Chapter 4) and is able to identify properties in tweets that may highlight that tweet as either being deemed useful or not useful to a user (Chapter 5).

However, we don't as yet know how useful these codes actually are. For instance is a tweet which includes a media link more useful than say a tweet with a trusted link, or is a tweet which contains a media link and has a higher lexical quality more or less useful than a tweet with a trusted link. To do this we have to have a test corpus on which to perform experiments.

In this chapter we describe a crowd sourcing study that we performed to create a test corpus that will allow us to generate weights (see Chapter 7) for the features we have extracted described in Chapter 5. The corpus we present is unique in that it gives all information needed to perform IR evaluation as described by Mannning (Manning, Raghavan, & Schütze, 2008)

In this Chapter we first describe the idea of weighting via an example explaining TF-IDF, followed by how we perform meaningful IR evaluation. From this we Describe the steps we have taken to produce this test corpus, and give stats as well as analysis regarding the collection of the data.

6.1.Weightings

The simplest of search engines are those that support Boolean queries: a document either matches or does not that match query. (Manning, Raghavan, & Schütze, 2008)In the case where corpuses are large, there may be a large set of results which match the query, this number could far exceed the number a human user could possibly search through.

As this set may be so large, we may wish to order these documents by some criteria. The search engine will compute for each matching document a score with respect to the query at hand.

6.1.1 TF-IDF Example

One of the most basic examples of weighting is the TF-IDF weighting algorithm. (Manning, Raghavan, & Schütze, 2008) This is calculated by obtaining the product of the term frequency for a given search term and the inverse document frequency for a given search term.

In the following section we will explain both term frequency and inverse document frequency, and give the final calculation to calculate TF-IDF .

Term Frequency

Term frequency, is a calculation where we calculate how many instances of a given term there are in a given document.

$$TF_{tem,document} = number\ of\ times\ term\ occurs\ in\ document$$

This is a 'bag of words' model where we do not care about the location of the terms within a document. We are just interested in the number of occurrences of each term within the document (the frequency).

In table 6.1 we have provided 3 example documents, if we try and calculate the TF for the term 'the' across all three documents we will arrive at the following results.

$$TF_{the,document1} = 2$$

$$TF_{the,document2} = 1$$

$$TF_{the,document3} = 1$$

The reason for this is, the term 'the' occurs twice in document 1, only once in document 2 and only once in the document 3.

However, normally we do not want the raw TF. As it may not necessarily be true that a document which has 10 occurrences of a term is 10 times more relevant than a

document that only contains one instance of the given term. (Manning & Jurafsky, 2015)

Log Frequency Weighting of TF

The idea behind TF weighing is that relevance goes up as the number of occurrences goes up. However, this may not be a linear increase, so we want to dampen this by utilizing a log function. We can use the following calculation to calculate a new weighting: (Manning & Jurafsky, 2015)

$$W_{term,document} = \begin{cases} 1 + log_{10}(TF_{term,document}), & if\ TF_{term,document} > 0 \\ 0, & otherwise \end{cases}$$

We have a conditional statement in the above equation as if there are zero occurrences of the term in the document the log of 0 is negative infinity, so we have to have conditions to handle this.

Document Number	Document
1	The cat sat on the mat
2	Porto is a city in Portugal, Lisbon is the capital city.
3	The mean squared error is a statistical measure.

Table 6.1 Example set of documents

Inverse Document Frequency:

Document frequency (DF) of a term is the number of documents that contain the term; regardless of the number of times the term occurs over the corpus or for each document. For instance if we look at our example documents in Table 6.1. The DF for the term 'cat' is 1 as it appears in document 1 only. The DF for the term 'the' is 3, as it appears in document 1, document 2 and document 3. Document frequency is an inverse measure of informativeness of a term (Manning & Jurafsky, 2015).

$$DF_{term} = number\ of\ documents\ that\ contain\ N$$

Inverse document frequency for a term is the total number of documents divided by the DF. We commonly use a log function to dampen the effect of IDF. The absolute

score may be seen as too strong a factor. (Manning & Jurafsky, 2015) The equation for IDF is given below.

$$IDF_{term} = log_{10}(N/DF_{term})$$

Where N is the number of documents (the size of the corpus). If a term occurs in every document IDF will be = 0, but as the DF decreases for a fixed size corpus, the value of IDF will increase.

IDF gives us the notion that rare terms are more informative than frequent terms. If we are querying for a very 'unique/rare/unusual' term, and the term appears only in very few documents across the corpus, we assume that the user would be highly interested in viewing these documents, as they are likely to be relevant to the user's query.

IDF takes the view that frequent terms are less informative than rare terms, whilst a document might contain a frequent term, it may not necessarily be the most relevant document.

So we wish to give documents with frequent terms positive weightings for documents matching a query, but lower than that of rare terms.

Calculating TF-IDF

The TF-IDF weighting of a term is the product of its TF weight and IDF weight.

$$W_{term,document} = \left(1 + log_{10}(tf_{term,document})\right) \times log_{10}(N/DF_{term})$$

According the Manning and Jurafsky TF-IDF is the best-known weighting scheme in for terms in IR. (Manning & Jurafsky, 2015) TF-IDF increases as the number of times a term occurs in the document (TF), but also goes up with the rarity of the term in the corpus(IDF).

To calculate the TF-IDF score of a document, we use the following calculation:

$$Score(q,d) = \sum_{t \in q \cap d} tf.idf_{t,d}$$

The score is calculated by summing the TF-IDF weighing where the term appears both in the query and the document.

Whilst the example given in this section is a weighting based on terms, we have multiple criteria which we have identified as reason for a tweet to be deemed useful or not useful to a user. By utilizing work in this chapter we hope to assign a weight to each of these codes, and to calculate how 'important' each code is to decided whether a document is useful or not useful via a machine learning based method called learning to rank, these steps are described in Chapter 7.

6.2 Performing IR Evaluation

To measure ad hoc IR effectiveness in the standard way we need the following things according to Manning: (Manning, Raghavan, & Schütze, 2008)

1. A document collection
2. A test suite of information needs, expressible as queries
3. A set of relevance judgments, standardly a binary assessment of either relevant or nonrelevant for each query-document pair.

In this chapter we explain the following steps seen in Figure 6.1 , that will allow us to evaluate our system against future systems. Please note that the final step 'Applying Learning to Rank' is described in the following Chapter of this thesis.

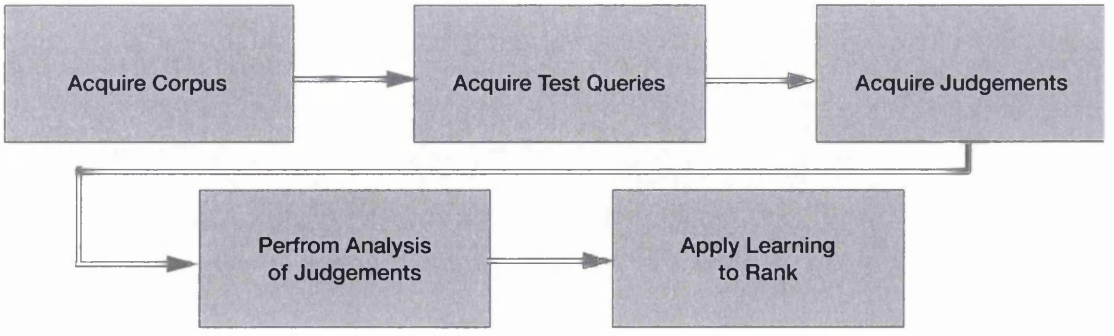


Figure 6.1 Steps involved in creating and using our Copora

6.3 Acquiring and Generating Test Tweet Corpus

In this section we explain the process we used to acquiring a suitable corpus and test query set for our evaluation of the system. We discuss existing evaluation corpora as well as the method we used to generate our own collection of test queries and a test corpus.

6.3.1 Existing Corpora

Whilst there were various corpuses available (see chapter 4.1), none satisfied the requirements to allow us to proceed with meaningful IR evaluation for the system we had created.

NIST offer the microblogging corpus, which deals with a very similar tasks, but is not an open corpus, that it does not allow researchers access to the user acquired judgments of the data. (Ounis, MacDonald, Lin, & Soboroff, 2011) Thus making the assessment of IR systems blind to the users who take part in the TREC microblogging track. Also rather than being open in what is meant by usefulness TREC assumes that the best result set for a given query is to "return the most recent but relevant information" (Ounis, MacDonald, Lin, & Soboroff, 2011), in this thesis we wish to investigate what is really the best information to return. This is what we hope this test corpus will deliver.

Due to not already having a corpus we have had to generate a corpus for this task. As we know we can not gather every tweet on Twitter (due to API and physical limitations), and get judgements for each tweet for a given set of queries, we have had to devise a system whereby we collect a subset of tweets which we think could be relevant to a set of pre defined queries, and test based on that data.

In the following sections we explain how we generated a list of queries to collect tweets, and how we generated the corpus based on these tweets.

6.3.2 Generating Queries

Ideally we would like a list of queries that have been generated by a group of users who have used a search engine to search over a microblogging corpus. Unfortunately

no publically avaliable corpus is available with this data. This is unfortunate, as it will not allow us to compare our system against a pre-existing system.

As no query corpus is available, we had to resort to creating our own. To do this we already had a set of queries which had been generated by users from the experiment described in Chapter 3, which involved the users performing search tasks based on three of the most common search tasks undertaken by people performing search over a microblog corpus. (Teevan, Ramage, & Ringel Morris)

Although we had an existing set of queries from Hurlock and Wilson (Hurlock & Wilson, Searching Twitter: Separating the Tweet from the Chaff, 2011), some of these queries were temporally relevant, as they are specific to certain events (BBC Proms in 2011) and the launch of products (iPhone).

For use in the experiments that we would be carrying out we had to modify these temporal aspects to be up to date with current events and products. Rather than the BBC Proms, we modified content so that it would be relevant to the Coachella festival, and the launch of the iPhone to launch of the iPad Mini.

We created a list of 48 unique queries from the set of original queries entered by users. The complete list of queries can be seen in Appendix A. These queries were not equally spread over each task in number.

Now that we had a test suite of information needs, expressed in the form of queries, the second things identified by Manning et al. (Manning, Raghavan, & Schütze, 2008), we need a corpus, and set of relevance judgments.

6.3.3 Our Corpus

Now that we have a list of queries, we need a corpus, which will allow us to query over. Whilst there are several twitter copra available, there were problems with using them. We wish to have system to compare our system against that would return results based on our queries.

The ideal situation would be to use the TREC corpus, but due to the lack of data regarding the corpus (judgments and queries) we felt that it was not suitable for the type of evaluation we wished to perform.

We wished to use Twitter's own search engine as a baseline. We therefore had to retrieve data from the same source. This meant grabbing data from Twitter, there are two ways to access this data. We could either record data from the sprinkler stream[1], or we could use the search API[2].

If we used the sprinkler stream, it meant gathering a large amount of data, however, only receiving an actual fraction of tweets. Alternatively we could query the search API, and record results that came through. The advantage of using the Search API is, that it will create a much smaller corpus, which will allow for quicker evaluation. However, it means we are relying on Twitter Search to return relevant tweets. Neither method will allow us to capture all tweets, however, we wish to compare usefulness, not relevance. So we have opted to use the second method (Twitter Search API).

Now that we have chosen a method to generate a corpus, we could generate the corpus.

We created a script that would access the twitter search API, then pick a query from our query list and return results for that query and save it with a timestamp to a file. This script ran via a cron job to run every hour, for a specified amount of time. Due to the size of the query list, it meant that we would exceed twitter's Search API rate limit. This meant we needed to split the script over several IP addresses and accounts, to get all the data.

We started to collect data on the 14th of April 2013, and finished collection on the 11th of June 2013. In total we collected approximately 5million query responses. Due to budgetry constraints we selected a subset of these queries and their responses to be

[1] https://dev.twitter.com/streaming/public

[2] https://dev.twitter.com/rest/public/search

featured in this corpus, with only 40,000 query, time of query, result tuple sets being selected. We split the type of task between the 40,000 as equally as possible.

Now that we have created a corpus (requirement number 1), and have a set of information needs (requirement number 3). We needed a set of relevance judgments.

This means we now have a set of queries, we also have a corpus of tweets. However, the tweets also have lots of other extra data, such as their position in a list of results for a given query, that was queried at a specific time. Allowing us to utilize this data as a baseline for any further evaluation. The only thing missing now to allow this corpus to be utilized for evaluation is human judgment of how useful that tweet is for a given query, when queried at a specific time. In the next section we describe how we gathered these judgments.

6.4 Generating Judgments

We needed to generate usefulness judgments for our tweets. However, we were constrained by several factors, time and available expert judges. In order gain the maximum number of judgments in the minimum amount of time we turned to crowdsourcing.

Crowdsourcing has been used in different areas of computer science and social sciences to obtain large amounts of data in short periods of time. It has been used for obtaining relevance scores (Alonso & Baez-Yates, 2011), as well as labeling tasks. (Lease & Yilmaz, 2012) (Welinder & Pietro, 2010)

Due to the constraints put upon us in terms of time and money. We decided selected just shy of 40,000 tweets to obtain judgments on. The tweets were selected to cover all three search tasks.

When deciding to choose which crowdsourcing platform to utilize, we were restricted in choice. Amazon's Mechanical Turk (AMT) is possibly the biggest and most well known of the crowdsourcing platforms. However, it is only directly accessible if you are a US based customer. Our next choice was to use Crowdflower. Crowdflower is one of the world's leading crowdsourcing service, with over 800 million tasks

submitted by over four million contributors. Crowdflower specializes in microtasking: distributing small, discrete tasks to many online contributors, in an assembly-line fashion. For instance, Crowdflower has been used to check hundreds of thousands of photos every day for obscene content. Crowdflower, has been used by other researchers, and offered us access to a large crowd of workers (It even allowed us access to AMT users through their platform, though at time of writing they no longer offer this service).

For each query and each result we need a minimum of three participants to acquire a meaningful DCG score (we discuss this in further detail later in this chapter) (Manning, Raghavan, & Schütze, 2008).

Participants were compensated based on the number judgments they successfully completed. We compensated participants $0.01 per judgment, we decided to compensate $0.01 per judgment based on works by Duncan Watts and Winter Mason (Winter & Watts, 2010) Though we also acknowledge that other authors have looked at how financial incentives improve quality of work performed by crowd sourced workers. (Horton & Chilton, 2010) (Buhrmester, Kwang, & Goslin, 2011)

As part of the process all participants had to agree to read and agree to an ethics and study design document, these documents can be found in the Appendix A.

As part of the work surrounding the crowdsourcing experiment we had to build a custom interface to deal with some interactions (such as consenting to the study and ethics) as well as getting the tweets to display as best as we could (to mimic twitters search result page). This involved using the Crowdflower markup language CML, HTML, CSS and JavaScript. Screenshots from the tasks can be seen in Appendix A and in Figure6.2 later in this Chapter.

6.4.1 Trusting Participants

In order to make sure participants did not 'game' the system, measures were put in place to detect whether a participant was to be trusted or not. We utilized Crowdflower's gold standard system as a mechanism to measure trust that a user is performing the task in a sensible manner.

CrowdFlower allows for gold standards to be put into the data, to check that participants are not bots. Participants will be scored for accuracy. If they fail these gold standard test, then they will be considered a bot and will not receive any compensation. More information about gold data can be found at http://success.crowdflower.com/customer/portal/articles/1365763-test-question-best-practices

We created two types of gold standard data. One where the query was a mathematical question consisting of the following:

What is (number from 0-10) with an operator (addition, subtraction, multiplication or division) followed by another (number from 0 to ten) followed by a question mark e.g.:

What is 2 x 10?

The responding tweet, then had the following text:

2 x 10 = 20

Replying with the question, and the answer. We had a similar set of questions where, the question would be:

What is the capital city of [Country]?

With the response text being:

The capital city of [Country] is [Country's captial city].

These were picked, as they are basic tasks, where answers could either be calculated quickly or could be looked up with great ease. When creating these test questions, we provided questions with the correct response, and incorrect responses. If the response

was incorrect, then the question was tagged as one of the answers on the not useful scale, if it was correct, then it was tagged as one of the responses on the useful side of the scale.

Users were shown a set of these test questions throughout the study. Users had to answer a minimum of 5 of these test questions before receiving payment. When answering these questions, they were assigned a trust score. This meant that if they incorrectly answered a certain percentage of these questions they would not receive payment, as they were counted as SPAM/Bots.

6.4.2 Participant Selection

Participants were selected based on their countries primary first language. It was desirable to have English native speakers for the basis of this task. We targeted users in the USA, Canada, United Kingdom, Australia and New Zealand. However, all users ended up residing in the United States, based on data provided by Crowdflower.

No age restriction was put on the users, and no prior experience of twitter was required of users as we can envision and have seen twitter results being embedded in non twitter specific search engines. However, to be able to sign up to Crowdflower you must be 18 years of age. So all participants were considered to be 18+[1]

Crowdflower acts a third-party provider tapping into the API of various crowdsourcing platforms, this means if we build tasks on Crowdflower, it will run via their payment system, or it could be outsourced to other crowdsourcing platforms such as AMT. We decided to allow our jobs to be sent to all available crowdsourcing platforms to enable the jobs to be completed in the minimum amount of time.

6.4.3 Usefulness Judgments

Now that we have described how we gathered our judgments we describe what the judgments consisted of.

[1] http://elite.crowdflower.com/index.php?view=terms

There are several ways in which we can generate a set of relevance judgments. Traditionally in IR judgments are seen as a binary indicator as either relevant or nonrelevant. However, relevance can be measured in levels. (Manning, Raghavan, & Schütze, 2008)

To properly evaluate a system, the test information must be relevant to the documents in the test corpus, and appropriate for predicted usage of the system.

According to Manning these information needs are best designed by domain experts. Using random combinations of query terms as an information need is generally not a good idea because typically they will not resemble the actual distribution of information needs. (Manning, Raghavan, & Schütze, 2008) This is why we have chosen to use actual query log data.

As mentioned in section 6.3.3 we collected approximately 5 million tweets, for our list of queries. It would have been too costly in terms of both monetary cost and time for us to have gathered judgments for all of these tweets.

So, we had to select a subset of these tweets. Our selection was made so that each task had as even coverage as possible. In total 40,000 tweet + search term pairs were selected, these were from search result ranging from the 14th April 2013 to the 22nd of April 2013. A maximum of 20 results per query were taken from each query.

We are aware that this corpus generation method will have introduced bias into our results, as we are basing our corpus on the accuracy of the Twitter search API returning relevant documents to us.

Manning does also note that a human is not a device that reliability reports a gold standard judgment of relevance of a document to a query. Stating that humans and their relevance judgments are quite idiosyncratic and variable. This is why we have decided to employ a large number of humans to perform a large task in hope of reducing this variability.

We have been able to see how idiosyncratic and variable judges are by utilizing a kappa statistic that creates a simple agreement rate for the rate of chance agreement. We discuss the agreement rate of our dataset later in this chapter.

Like TREC and other evaluation systems, we have adopted an ordinal notation of usefulness with documents divided into four classes distinguish documents being not useful from those which are very useful.

There are several ways in which to traditionally perform relevance for a retrieval system, depending on how we mark documents as useful or not. We can also look at how we evaluate a list of unranked results, or ranked results. We are most interested in ranked retrieval results as we hope to bring the most useful tweets to users. Below introduce two of the common measures which allow us to the do this.

Mean Average Procession(MAP)

Possibly the most common measure is mean average precision (MAP) (Manning, Raghavan, & Schütze, 2008), which provides a single figure measure of quality across recall levels. Results from MAP have been shown to have especially good discrimination and stability. (Manning, Raghavan, & Schütze, 2008)

For a single information need, average precision is the average of the precision value obtained from the set of top k documents existing after each relevant document is retrieved, and the value is then averaged over information needs.

For a single information need, the average precision approximates the area under the interpolated precision recall curve, and so the MAP is roughly the average area under the precsion-recall curve for a set of queries.

DCG and NDCG

Discounted Cumulative gain and Normalized Discounted Cumulative Gain, are measures that have seen increasing adoption, especially when employed with machine learning approaches to ranking. NDCG is designed for situations of non-binary notions of relevance. (Manning, Raghavan, & Schütze, 2008)

Due to there being various levels of usefulness we have decided to use NDCG as the primary evaluation of our system.

As we have chosen to use NDCG and we have a large dataset with paired queries, we needed a way in which to collect usefulness judgments for each tweet and its corresponding query. To do this we implemented the crowd sourcing tasks to gain usefulness scores.

In the next Section we describe the Crowd Sourcing Experiments we ran in detail.

6.5 Generating the Corpus

Over the course of the Ph.D. we generate 4 different corpuses. This was due to low kappa scores. We discuss the corpuses we generated chronologically and discuss the reasons for generating the 4 corpora and utilizing one in the end.

It is worth noting that the layout of the interface did not change, nor did the amount of questions per page change (set at 5) over all corpus generation, the only thing that changed was the wording to do with the answers users could give, as well as the amount of answers to pick from. All units were judged by three trused contributors from the Crowdflower platform in these experiments.

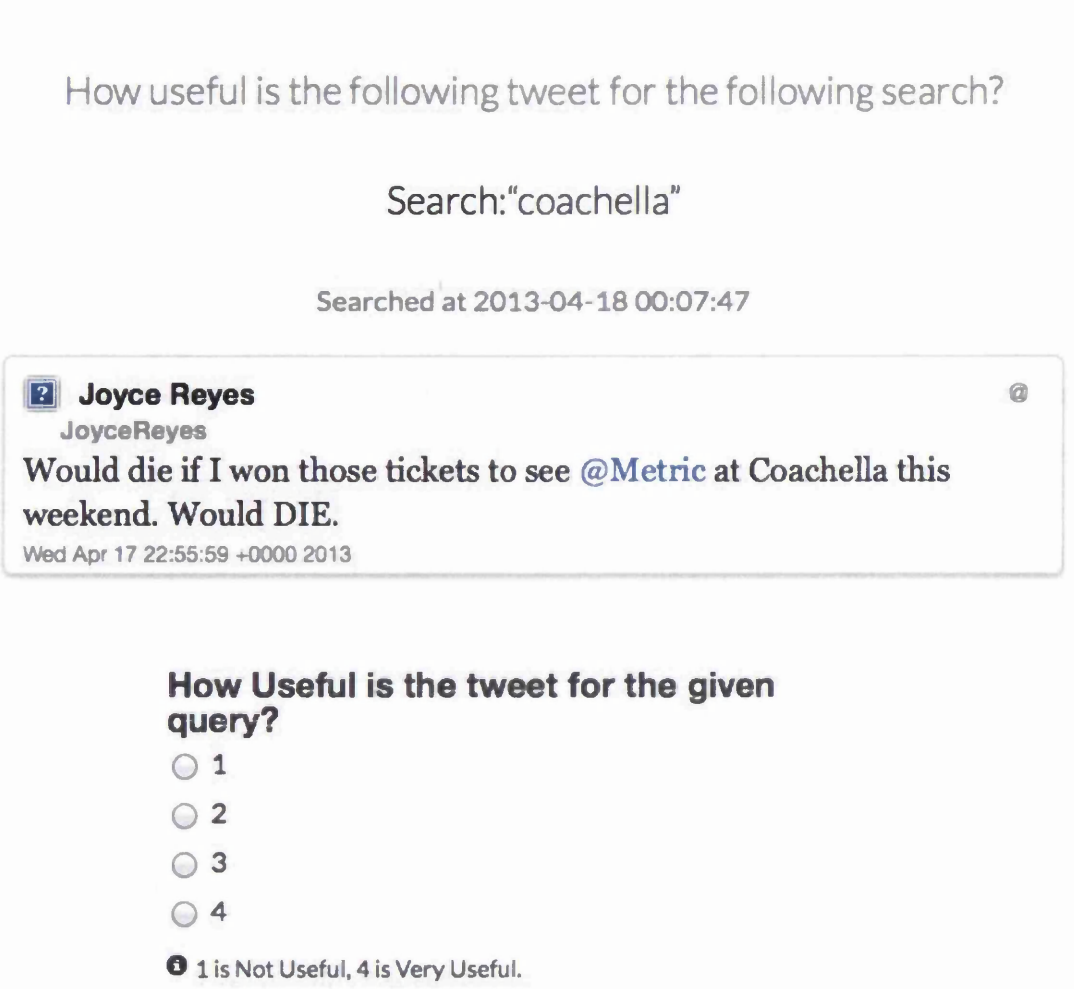


Figure 6.2 Screenshot from Crowdsourcing study – Corpus 1 (215802)

Please not that even though there is a missing image at time of running the experiment images were present. Also the @symbol in the top right of the tweet would be next to the user's twitter alias in gray (this is an error to do with the image missing).

6.5.1. Corpus 1 (215802)

Experiment 215802 was the first crowd sourcing study we undertook. It consisted of 39994 units, of which 4998 were test questions and 105001 judgments were made. The wording used was based on the literature presented in Introduction to Information Retrieval (Manning, Raghavan, & Schütze, 2008).

Participants were presented with a scale of 1 to 4. Where 1 was – Not Useful and 4 being Very Useful. 2 and 3 were left open to interpretation to the participants.

To evaluate the judges agreement we used a Krippendorff's alpha (Krippendorff,

2013) which is used to measure inter-rater reliability. The alpha score, scored was 0.0945349960529 rated as low agreement according to Fleiss and No agreement according the Landis and Koch.

Based on this low score we decided to look at our experimental setup. Each piece of data had three judges judging it. No judge was seemed to be gaming the system. We examined the results and even ran the experiments taking out the top contributors; this made little difference to the alpha score. Indicating no one user of the top 5 contributors was skewing results.

As a result we looked at the wording used in the experiment and decided to change it to see if it mad a difference. This then led us on to conduct experiment 390484.

6.5.2 Corpus 2 (390484)

Experiment 390484 was the second study conducted. It was conducted to see if the openness of experiment 215802 was creating disagreement between users. Due to budgetary constraints we used a subset of the data used in the 1st experiment. With 17422 units, 2425 test questions generating 45017 judgements. Again a low alpha score was achieved (0.0665999143908) indicating even lower agreement.

Participants were asked to How useful is the tweet for the given query, with 1 - Definitely Not Useful 2-Maybe not Useful 3-may be Useful 4-Definately Useful

Based on this, we ran another experiment keeping with the same language as experiment 1, though giving options 2 and three an explicit answer.

6.5.3 Corpus 3 (392186)

Experiment 392186 was also in response to the low kappa statistic found in experiments 1 and 2. This time an even lower alpha score was retrieved (0.0419000427235). Indicating that possibly four possible options was too much.

The participants were asked the same query, with the following options: 1-Not Useful 2-A Little Useful 3-Quite Useful 4-Very Useful. Instead of giving users 4, options we also ran a binary experiment (see corpus 4)

6.5.4 Corpus 4 (414908)

Experiment 414908 was the final experiment conducted, due to all the previous experiments receiving low kappa scores, we decided to investigate whether our scale was too sparse. The participants were given a binary option in this task rather than the 4 point scale in the previous experiments with 1-Not Useful and as 2-Useful. This was run as a larger experiment with 3994 units, 4998 test questions and gathering 105055 judgements. This returned an alpha score of 0.132293455368 achieving the highest alpha score.

Whilst this offered the highest alpha score of all corpuses generated it was not what we wished to have in terms of a usefulness score.

6.5.5 Further Analysis of Corpora

As the alpha scores were so low, we also experimented with grouping judgments, for instance in Corpus 1, we combined scores 1 and 2 to make not useful, and 3 and 4 to make useful, thus turning it into a binary task, however no great gains were found, we also tried groupings of 123-4 and 1-234.

From looking at the alpha scores from these experiments we can only conclude that judging a query, response for a particular period in time is a very subjective task, and thus may make defining a tweet as either useful or not useful for commonly found search tasks harder than originally though.

We release all corpuses in the following as two files is CSV (Common Separated Value) format, the corpora file, which contains the key data which adheres to the Twitter terms of service. As well as Query File, which is a copy of the queries found through the experiment. The structure can be seen in the paragraphs below.

Corpora File Format:

Query_time, Query_Id Tweet_Id, Rating_for_NDCG, Judge_Id, Golden_Data

Query File Format:

Query Id, Query

As well as our judged corpora files, we also release a copy of the original twitter search results pages, in a format which adheres to the Twitter terms of service. This will allow users to perform analysis between both ours and twitter's search engine.

6.6 Summary

In this chapter we have described the methodology and reasoning that we implemented, constructing four crowd sourced IR evaluation test corpora. Through analyzing the inter-rater agreement we have discovered that usefulness is a very subjective measure based on the judgments we have collected.

Despite the low inter-rater agreement, we have chose to explore if any of the codes we found in Chapter 3, can be of use in detecting useful or not useful tweets. We show how learning to rank can be utilizing in the next Chapter to weigh codes based on the data collected in this Chapter.

Chapter 7: Ranking Factors of a Useful Tweet

7.1 Introduction

In this Chapter we aim to gain insight into how features in tweets, can make a tweet be deemed useful or not to some one performing search over a microblogging dataset. Most importantly though in this chapter we wish to understand exactly how important each of these features are.

Using the corpora and judgments provided in Chapter 6 we can create optimal ranked lists based on these judgments to optimize NDCG scores. By understanding what features are in these lists, we can then apply a machine learning technique called learning to rank, which will calculate boosting scores for these features, telling us how important each feature is.

In this chapter we firstly introduce the notion of learning to rank, and some work that has used learning to rank to optimize retrieval models, we then describe how we performed learning to rank on our data and the scores we received for the features.

7.2 Learning to rank

There are many ways of ranking documents in Information retrieval such as cosine similarity (Manning, Raghavan, & Schütze, 2008), TF-IDF (Manning, Raghavan, & Schütze, 2008)(an example of this is given in Chapter 6), proximity (Manning, Raghavan, & Schütze, 2008), pivoted document length normalization (Manning, Raghavan, & Schütze, 2008) and pagerank (Brin & Page, 1998) to name a few ways.

There are also many ways in which we can classify documents using supervised machine learning classifiers such as naïve Bayes (Murphy, 2012), kNN (Murphy, 2012) and SVMs (Murphy, 2012) (Manning, Raghavan, & Schütze, 2008).

Learning to rank is a way of utilizing machine learning to rank documents displayed in search results. It is also referred to as machine-learned ranking or machine-learned relevance.

Many major search engines already employ learning to rank, to rank their search results. Amit Singhal admitted Google employed over 200 features to calculate how relevant a document was and where it was to be placed in a search engine results page. (Hansell, 2007)

Learning to Rank has been used in Twitter specific retrieval research. It was used by McCreadie and Macdonald who looked at increasing relevance in microblogs search by looking at the content of links in tweets. (McCreadie & MacDonald, 2013). Naveed et al also used a generalized linear regression learning to rank model to give weights to features they had identified (Naveed, Gottron, Kunegis, & Alhadi, 2011).

Learning to rank is usually performed by supervised, semi-supervised or reinforcement learning. In our case we will be using a supervised approach, with training data consisting of data gathered in the previous chapter. Typically binary or ordinal scoring is used in training data to show how relevant each item is for a given query. (Manning, Raghavan, & Schütze, 2008) In our case we will be using an ordinal score on a sale of 1 to 4 as setout in the previous chapter.

The purpose of learning to rank is to find a function that produces a permutation of items, which is similar to the ranking of the training data. By doing this we can find which features are more important or less important to finding what makes a tweet useful or not useful. This will allow us to perform 'boosting' (Manning, Raghavan, & Schütze, 2008) of results to display the most useful tweets at the top of a search results page.

7.3 Learning to Rank Useful Tweets

We have taken the approach of a supervised learning approach to learning to rank. This involves us generating a training set of data. By using the data generated in chapter 6, we can obtain a preferential ordering for tweets given certain queries by ordering to maximize NDCG. We utilize the usefulness scores in chapter 6, as a way of ordering documents. We have averaged the usefulness scores between judges to produce a target value, of where we would like documents to appear in a search engine results page. Several documents may share the same value target.

Each document is put through our search system whereby each document has its features extracted based on the codes described in chapters 3 and 5.

After each document has been given a target value for each of the queries where it appeared in a search engine results page, and each document has its features extracted this data in collated and synthesized into a training file.

The file takes the form of that SVM^{rank} uses (Thorsten, 2006).An example is given below

```
<line> <target> qid:<qid> <feature>:<value> <feature>:<value>...
<feature>:<value> # <info>
```

where each of the inputs are of the following values

```
<target> .=. <float>
<qid> .=. <positive integer>
<feature> .=. <positive integer>
<value> .=. <float>
<info> .=. <string>
```

An example input file can be seen below

```
3 qid:1 1:1 2:1 3:0 4:0.2 5:0 # 1A
2 qid:1 1:0 2:0 3:1 4:0.1 5:1 # 1B
1 qid:1 1:0 2:1 3:0 4:0.4 5:0 # 1C
1 qid:1 1:0 2:0 3:1 4:0.3 5:0 # 1D
1 qid:2 1:0 2:0 3:1 4:0.2 5:0 # 2A
2 qid:2 1:1 2:0 3:1 4:0.4 5:0 # 2B
1 qid:2 1:0 2:0 3:1 4:0.1 5:0 # 2C
1 qid:2 1:0 2:0 3:1 4:0.2 5:0 # 2D
2 qid:3 1:0 2:0 3:1 4:0.1 5:1 # 3A
3 qid:3 1:1 2:1 3:0 4:0.3 5:0 # 3B
4 qid:3 1:1 2:0 3:0 4:0.4 5:1 # 3C
1 qid:3 1:0 2:1 3:1 4:0.5 5:0 # 3D
```

Where 1A is greater than 1B, 1A is grater than 1c, and 1B is greater than 1C and so on so forth.

Code Number	Code Name	Boosting to be Applied
1	direct recommendation	-0.92958492
2	Language score	-0.13046047
3	lexical quality	-0.13046047
4	link actionable links	-0.050322037
5	link dead links	-1.1377115
6	link media links	*
7	link trusted links	0.37946385
8	link untrusted links	*
9	link useful links	-0.68356311
10	part of conversaion	*
11	Personal experience	-0.99849784
12	Question and answer	0.08155752
13	Retweets	*
14	sentiment fisher	0.47088608
15	sentiment naïve bayes	-0.4607574
16	Specific information presence of nouns	-0.77061963
17	Specific information presence of price	0.24662203
18	Specific information presence of time	0.16819793
19	SPAM wang	*

Table 7.1 Showing boosting scores to be applied based on linear kernel function of SVM learning to rank. * denotes that due no instances were detected in the dataset and all values were set to 0 in the corpus.

We then run the data through a learning to rank algorithm that provided us with boosting scores. Although there are many variants of learning to rank algorithms we have chosen to adopt a linear learning function, as provided by the SVM^{rank} tool (Thorsten, 2006)

By running the data through the learning to rank program we are given an output that computes feature gains. Feature gains give us hints to how much useful the features used in the training set are. This enables to boost features based on these scores, thus enabling us to re-rank the data returned from the default elasticsearch output.

7.4 Learned Weightings

In this section we present the learned feature gains for our codes described in Chapters 3 and 5. This is based on the 19 features, 392, rankings and 2565 results utilizing corpus 1 (215802). We used the default linear classifier provided by SVM^{rank}, with a linear kernel function are shown in table 7.1

7.5 Analysis of Weights

As we can see from table 7.1 there appears to be a lack of some of the codes in the test corpus, or that the programs detected a score of 0 for all instances, these are denoted by asterisks in table 7.1.

Perhaps unsurprisingly dead links is the feature with the biggest boost .This shows that links that are working are possibly the biggest useful feature in tweets being deemed as useful. Personal experience and direct recommendation are the biggest boosts to follow, however boost in the opposite direction as expected based on the work carried out in Chapter 3.

Perhaps some of the most interesting boosts are that of the specific information and useful links. It is of no surprise that presence of price and a mention of time is considered to be more useful, however if a proper noun is detected that tweet is to then be boosted negatively. This could perhaps be put down to the POS tagger incorrectly tagging words as proper nouns when in fact they are not.

As for useful links being less useful it maybe down to links not being able to be scraped so then a poor lexical score is given, thus affecting the useful link score.

Not all boosts were as one might expect based on work in Chapter 3. However, this may also be down to the low agreement scores between judges in Chapter 6, causing erratic boosting.

7.6 Weights learnt from other Datasets

Due to some of the unexpected results found from running SVM^{rank} over the 215802 (802 – this is a shortened name using the last 3 characters of the corpus id) corpus. We decided to run the SVM^{rank} over all of the corpuses we had collected during the crowdsourcing stages of this work. We ran SVM^{rank} four times in total.

We present the results of this process in the Table 7.2 (including the results from the 802 corpus).

Dataset Name	802	484	186	908
Dataset Size	Larger	Smaller	Smaller	Larger
Judgment Scale	Non Binary			Binary
Notes	Best Alpha for non binary judgments			Best overall alpha
Code Name	Boosting to be applied based on dataset			
direct recommendation	-0.92958492	*	*	-0.029490978
Language score	-0.13046047	-0.057084251	-0.12709083	-0.0061487784
lexical quality	-0.13046047	-0.057084251	-0.12709083	-0.0061487784
actionable links	-0.050322037	-0.2949864	-0.37272727	-0.18888389
link dead links	-1.1377115	0.006418766	0.13097546	- 0.54472011
link media links	*	*	*	*
link trusted links	0.37946385	0.031602722	0.26761159	0.14894152
link untrusted links	*	*	*	*
link useful links	-0.68356311	-0.086665802	-0.26475281	-0.14595798
part of conversation	*	*	*	*
Personal experience	-0.99849784	*	*	-0.032967035
Question and answer	0.08155752	0.72931153	0.4982377	0.47662315
Retweets	*	*	*	*
sentiment fisher	0.47088608	-0.12611616	-0.020621078	0.074158199
sentiment naïve bayes	-0.4607574	-0.31410402	-0.16607563	0.057382833
SI presence of nouns	-0.77061963	0.028215462	0.27272728	0.46349153
SI presence of price	0.24662203	-0.19917543	0.25409713	0.28403515
SI presence of time	0.16819793	-0.21735726	0.29046082	0.011222886
SPAM wang	*	*	*	*

Table 7.2 Showing boosting scores for all for corpuses generated.

From looking at the results in table 7.2 we can instantly see that the size of the corpus has some effect on the boosting scores. We can see that direction recommendation and personal experience codes are not assigned scores for corpuses 484 and 186. This is due there being no instances of these codes within these corpuses. So the classifier is unaware of them.

This is not purely due to the size of the corpus, but is down to the amount of coverage of codes (i.e how many times do they appear in the corpus). Ideally we would like to as even coverage of all codes as possible, with as many instances as possible, to help train a good model. Alonso et al discussed corpuses sizes and the issues surrounding corpuses size and coverage, when it comes to judgment tasks. (Alonso, Marshall, & Najork, 2013)

Like many of the codes we extracted we see opposite scores to the ones we thought we would have seen based on our findings in Chapter 3. One examples of this is direct recommendations, this was a positive code in chapter 3. However, when when using SVM^{rank}, we are presented with a negative boosting.

Boosting such as that described in the pervious paragraph could be due to a number of reasons. Initially it could be due to poor detection of the code itself, if we think of this as a classification task, we may aim for precision when detection codes, over recall depending on what kind of code we are searching for. Thus we actually under or over sample on the code we are targeting.

Alternately it could be due to the kernel function utilized in the classifier. SVM^{rank} utilizes a linear classifier. We are using a high dimensional space onto which the classifier must find an optimal hyper plane. Due to this kernel being a hyper plane, and not a more complex kernel function, we may be under fitting our model, thus giving us non representative outcomes.

In chapter 3 we noticed that combinations of codes, provided either positive or negative reasons as to why tweets were deemed useful. In chapter 3, we provided a brief analysis as to some of the combinations which made tweets either useful or not useful. In terms of the classification we created scores for individual codes, however, we did not provide scores for combined codes. E.g. personal experience and direct recommendation. Perhaps from taking this naïve approach we missed important combinations of codes.

As previously mentioned we know that corpus bias can effect the outcome of performing learning to rank. We can see from table 7.2 that when a code is not present in a corpus it will not have a score associated with it, as the machine can not make a judgment based on something which is not explicitly represented in the corpus. As well having 0 instances of a code, having a small dataset may hamper the boosting scores, as there are not enough samples to create a generalized model for all tweets. This is true of all supervised learning approaches.

Work by Alonso (Alonso, Marshall, & Najork, 2013) found as we add more judges to a task we increase the coverage of the topic we are trying to understand by adding more perspective (in their case interestingness), but in doing so we are also decreasing consistency. Ultimately Alonso recommends using a small number of very experience judges rather than a large number of diverse judges.

We had three judges, judge each data point. However, these judges were not expert judges, and we did not have three judges in total, judging the datasets. Each dataset had a large number of judges, each judge was contributed to a small proportion of each set, allowing for a decrease in consistency and increase in variability. Perhaps due to the variability we see different scoring.

When looking at the scores obtained for Questions and Answering we find that this it he most highly boosted score across three of the four datasets, indicating that there may be some consistency between judges for singular codes in terms of what makes a tweet useful. We also find that dead links has the lowest boosting score for both of the larger datasets, meaning unsurprisingly that tweets with links that are alive are a positive trait.

In terms of specific information, we see that in all but two cases (corpus 484), specific information is considered to be a positive boost. When looking at this, further, we notice that price is considered to be a negative, for this corpus, this could be due to the fact a lot of the tweets in the corpus especially to do with the task concerning finding specific information may be labeled as not useful. This could be due to there being a lot of "spammy" messages along the lines of "good news ipads are now only £100",

and thus may be labelled towards the not useful side of the scale, which in turn could mean that the classifier learnt whenever a price was detected, it learnt that this tweet is not useful / spam because price is being attributed to spam tweets.

We see that for both of the smaller corpuses, sentiment play a negative effect on a tweet considered to be either useful. Based on our coding in Chapter 5. Tweets with a negative sentiment are given the score of -1, and tweets with a positive score are given a score of 1. For the smaller datasets, we see that boosting is a negative floating point number, meaning this will reverse the value of the sentiment. Thus meaning that negative sentiment is positive, and positive sentiment is negative. Again similar to the specific information, we may find that spammy tweets overly utilize positive sentiment. Otte et al. (Ott, Cardie, & Hancock, 2013) have observed how sentiment richness has been able to be an indicator of spam. We believe this is the underlying cause of this scoring.

The lexical quality of a tweet consistently had very little influence on how well a tweet was deemed useful or not. This is perhaps dues to the nature of the medium, where people expect the language not to be perfect, and due to the constraints accept some errors. Rello and Baeza-Yates commented on how well lexical quality was over the twitter dataset compared to other online social networks. (Rello & Baeza-Yates, 2012)

Links had a very interesting story to tell when it came to weightings. If a tweet contained a trusted link this seemed to be a consistently good indicator of a tweet being useful. However, actionable links some had a negative effect, this may be due to the content of actionable links, where originally we saw an actionable link as an online store, or tool, we may find that actionable links are actually asking for details, and users just want to fulfill an informational or transactional need as quickly as possible. So have more forms on a page is seen as a negative trait. Whilst this code had a small boosting effect in corpus 802, for the remaining corpuses it had a more significant boosting. Being the most negatively boosted code for corpus 186.

We also looked at a link's usefulness, this was looking to see if a link was an authoritative source, we inferred that authoritative sources had a high lexical quality as discussed in chapter 5. A score of 1 mean a link's lexical quality was perfect 0 meant it was just jibberish or the link was dead. We find consistently across the boosting scores in table 7.2 That a links "usefulness" was to be deemed a negative indicator of usefulness. After apply SVM, we went back and tested the coding for this code, and found that keywords such as iPhone iPad and Coachella were not detected as words correctly spelt, and due to the tasks being concerned with the above keywords, we believe that expanding our dictionary to taken in terms such as those relevant to the task could have played a small part in improving this. This code suggestests that imperfectly written content or content containing unique terms, maybe more useful than higher lexical quality content. This may be due to the nature of the content e.g. reviews, opinions etc. However, we come onto a very important point in the next paragraph.

Unfortunately, one thing we were not able to track was if judges actually clicked on the links when judging tweets. We don't know if they knew what lay behind the links they were presented with, meaning some of the weightings to do with links may be biased to how people interpreted the links that were shown to them, rather than the content behind the link.

7.7 Summary

In this Chapter we give a brief overview of how learning to rank has been applied to other twitter retrieval models. We also introduce how we have performed learning to rank, and have presented the results of a linear learning model applied to the features we extracted in Chapter 5. We discussed some of the interesting boosting values shown in Table 7.1

In this following Chapter we conclude this thesis giving an overview of how we have attempted to achieve our research contributions and give a brief overview of the work covered in this thesis.

Chapter 8: Conclusions

8.1 Introduction

In this section we conclude the work this thesis covers. Firstly we revisit the research contributions we aimed to fulfill in Chapter 1. We then discuss how well we achieved at fulfilling these research objectives. We then discuss the limitations, commenting on limitations we had and how the work may be limited in representing the population as a whole. In the limitation and discussion s section we also discuss how changing our approach could have benefited our analysis. At the end of this chapter we discuss future work that could be conducted to further knowledge in areas relevant to this research.

8.2 Research Contributions Revisited

In Chapter 1 we set out four research contributions we wished to achieve.

- To understand what factors make a tweet useful to people performing searches of microblogging data
- Develop a robust framework for indexing and retrieving large amounts of microblogging data in a timely fashion
- Create a test corpus to allow us to compare our system against others
- To produce a system whereby we can index large datasets, and retrieve data in a timely fashion, and automatically whether a tweet is to be deemed useful or not for a given query

The first objective we had was to gain an understanding of what factors make a tweet useful to people performing searches on microblogging data. We addressed this using a qualitative and quantitative approach in chapter 3. Through this approach we found reasons as to why a tweet may be deemed either useful or not useful to someone searching a Twitter corpus based on the three common search tasks performed on microblogging data.

By exploring and utilizing the architectures described in chapter 4, when then tried to programmatically identify features in chapter 5 based on the work carried out in

chapter 3. From there we wished to evaluated how important each of the programmatically programmed features were in chapter 7, based on a test several corpuses we had created in chapter 6.

Whilst we identified a list of reasons as to why tweets may be deemed useful or not in chapter 3, there seemed to be some conflicting evidence in chapter 7, as to what made tweets either useful or not useful to someone performing search over a microblogging corpus.

This may be due to a mixture of the subjective nature of the person providing judgments when creating our crowd sourced corpuses, the method utilized when gathering content for corpuses, and the way in which codes were programmatically extracted. We discuss this in greater detail later in this chapter.

The second research objective was to build a scalable robust framework which would allow indexing and retrieving of search results in a manner that was appropriate for users using the system. This was demonstrated in the work described in chapter 4.

We have presented a batch indexing system. At the time of research and development there were no systems mature enough which we could utilize to perform near real-time indexing of large social streams of data. Whilst this would be favorable unfortunately this was unfeasible at the time. Since the development of this system, technologies such Storm[1] and Flink[2] have come about allowing for this type of near-time real time computation to occur.

We have described a method in chapter 6, that describes a test collection which allows researchers to perform microblog search performance against our system but also against Twitter's search engine based on the results we originally gathered.

[1] https://storm.apache.org/

[2] https://flink.apache.org

As mentioned in chapter 1 we are seeing more and more interest in social media, it is becoming more and more prevalent and part of our everyday lives. Research surrounding online and mobile social networks is continuing the increase. We are generating more and more content every day and a large amount is being generated by social networks. The ability to index and surface valuable content to users is a form which is easily digestible is a very hard and interesting problem. We have attempted to answer this question in this thesis.

Whilst performing this work we have seen changes in behavior of how people have utilize social networks as well as seeing new social networks appear and disappear. We are collecting more data now than ever, and a lot of the time it is richer data. Telling us where, where and what something is about, as well as some kind of content. We can see this even with Twitter by looking at Raffi Krikorian's white paper on what meta data a tweet contained in 2010 (Krikorian, 2010) to what the payload looks like now (Twitter Inc.).

Utilizing the is rich data, we only hope to see search across web, mobile, internal product search to improve. Delivering more relevant / useful / interesting / fun content to the user. We are seeing massive jumps in terms of advancement with new machine learning techniques such as deep neural networks to help build better learning models, and to better help understand data. We hope this work can contribute to building better mechanisms for search and go on to help people find useful results quicker, and easier. Even if this work helps to define features as to what makes tweets useful or not.

8.3 Limitations & Discussion

There have been limitations as to what we could achieve with this work. The research that we conducted used a lot of resources, not only in terms of computing power, but also in terms of monetary expense.

Ideally we would have like to have gathered more judgments per datapoint. We would have liked to have attempted this in the hope that this may have reduced the amount of variation between judges and increased the alpha score when creating the corpuses

in Chapter 6. Although, by adding more judgments we may in fact actually increase variation between judges, allowing us enforce the point that usefulness is a very subjective matter, and varies with the judges' own interests and proclivities.

As well as increasing the amount of judgments per data point. It would also be useful to have obtained more datapoints. As the corpus increased in size (see table 7.2), we started to see codes being activated, as well as codes generalizing.

We believe that due to the content which may have been collected in the specific information task, it may have caused several codes to have "flipped" due to the content surrounding these tweet+query pairs. On further manual inspection a majority of these tweets seemed to be spammy in content. They did however, correctly detect other codes.

Further more, this leads us to believe we need a better way of detecting spam based on a tweets content, not in terms of duplication, but it terms of trustworthiness or motivation behind the message. E.g. unsolicited messaging or advertising fake / scam deals. Rather than just looking at the network structure or elements such as links, hashtags, mentions contained within an tweet.

As well as increasing the amount of judgments made upon each data point, we believe it would have been useful to have run the experiment in a similar way to Alonso's work. (Alonso, Marshall, & Najork, 2013) Allowing for two types of judges, both a mix of expert judges and crowd sourced workers, with the exception of allowing non twitter users the chance to become a judge. We feel that having users who are not necessarily knowledgeable about Twitter but who could come into contact with Twitter based content is an important design decision.

One of the problems faced whilst undertaking this work and was also faced by (Alonso, Marshall, & Najork, 2013) was that of low agreement based on the Krippendorff alpha scores we retrieved.

We tried modifying the rating schema, and looking for users who were 'gaming' the results, and found to no avail, that judges were unable to agree for a random tweet+question its usefulness (depending on the scale used).

Like Alonso we also found that a binary judgments provided us with the highest scoring alpha (0.132), though was not sufficient enough to warrant real appraisal. We tried modifying the scales (using different groupings) used within the mutli-level judgment corpus to see if we could elicit a higher alpha score. Which ultimately led to us attempting the binary scale of either useful or not useful.

However, having said this one of the biggest limitations of this work is how applicable it is to the population as a whole. We have been working with subsets of data throughout this thesis. Unless you are retrieving the firehose stream, you don't really know how much of the actually twitter feed you are receiving, making it hard to estimate how representative your sample is. Only a few companies have access to this data resource, and from the statistics based on tweets per day, we would not be able to handle storing and processing that amount of data in the current lab environment we work in.

Twitter is a multilingual, multicultural, and a global network. Through out this thesis we have concentrated on English language content and US and UK centric events. It is hard to say if the findings in this work will translate over to other cultures, languages or communities with in the twitter network.

Whilst this work has been set in the context of Twitter, we have seen strong relations to work carried out in the field of information retrieval as a whole. Work described by Barry and Schamber (Barry & Schamber, 1998), bore striking similarities to work we carried out in Chapter 3, even though both Barry and Schamber were looking at relevance. However, we did notice some subtle differences, though this was most task specific. This is not to be seen in a negative light, but it helps us work towards a better understanding of how we help improve the user's experience, by surfacing relevant and useful information to the user. Finding these similarities helps to further define how we can best surface information to a user.

As well as seeing similarities to Barry and Schamber's work, Spink et al. (Spink, Greisdorf, & Bateman, 1998) state "The measures of usefulness, ... and satisfaction measure other important factors that users may employ in making relevance judgments and are sometimes used in research as an alternative way to define and measure relevance", this may explain why we may have seen such similarities between our work and others.

8.4 Future work

As part of this work we have made several contributions in an attempt to understand what makes tweets useful to users performing search over microblogging data.

We have seen that the results can in some cases be conflicting and very subjective. Further work building looking at how variations of judges, corpus size and corpus content would be beneficial to this, helping to confirm or either deny the results we have found.

As well as either confirming or denying our findings. To more thoroughly understand behavior to do with people finding information useful, we could attempt to profile users, in an attempt to see if certain types/classes of users find certain types of information or traits found within a tweet useful. If this is found to be true, taking a multi model approach and building a recommendation engine could be beneficial to the user.

We believe that the major motivation behind future work should be to try and discover the reasons as to why our Krippendorff's alpha score is so low, and to try and iterate on ways to increase this alpha score, till we obtain a moderate agreement. This could be done by taking inspiration from Alonso's work. (Alonso, Marshall, & Najork, 2013)

In chapter 7 we performed learning to rank utilizing a SVM with a linear kernel. It would be of interest to see if other kernels (for SVMs) and other learning to rank algorithms could be applied to reduce error rate, though being aware of not over fitting this model.

This leads on to something we did not cover at the end of this work, human testing of the search engine. We started with a very humanistic approach at the start of this thesis, looking at how humans classify tweets as either being useful or not useful for a search task, we then started to build technology to automatically extract these features, from there we utilized humans again as judges, in the hope of building ranked datasets. Once we had these ideally ranked datasets, we then again turned to technology to quantify weights for these factors. However now we have obtained these weights we have not actually seen if they provide better results from a human perspective.

It would be very interesting to compare whether the weights we found provide better results to that of another search engine, which does not use these weights. There is however a problem, we do not have entire history of tweets, nor could we conceivably collect and process this amount of data. So this evaluation would have to be restricted to a closed dataset. This could simply be run as several A/B test to compare which system provides better results from a purely human point of view. We suggest having both a mix of non-expert and expert assessors, as we believe that non-expert users will likely use the system and therefore be interested in the results.

Near real time processing frameworks have come along way since starting this work, with no near-real-time big data processing frameworks available at the start of this project we now have several to choose from. From both an engineering perspective and user experience it would be very interesting to have near-real time solution to this problem. However, from a purely engineering perspective this is a very hard and complex task. From a user perspective it allows us to deliver up-to-date temporally relevant content which is one of our codes found in chapter 7 and was also seen as a reason for relevancy between Barry and Schamber. (Barry & Schamber, 1998)

Appendix A

List of Queries to Build Test Copora

Study Plan for Crowd Sourcing Study

Ethics Approval Document

Research Consent Form

List of Queries Used to Generate Corpora in Chapter 6.

#Mini	dinner in London
buy iPad Mini	London cafe
cost of iPad Mini	London food
#iPad Mini	cafes in London
#iPad	London
London cheap restaurant	#foodie London
eating in London	#London London
Restaurants in London	coachella programme
"good food London"	coachella & Hotels
London cafes	coachella review
London cheap eating	coachella california
Restaurants in London%2C London	coachella events
eat lunch in London	coachella tickets
#London restaurant reviews	coachella tickets book
cafe London	coachella reviews
London restaurants	coachella schedule
best London restaurants	coachella June
London eating out	coachella location
eating out in London	tickets coachella
London lunch	price of coachella
good restaurant in London	coachella
#London restaurants	#coachella
London restaurant	coachella who is going
great lunch London	coachella so far

Study Plan

Research Project Title

Advance Information Retrieval: Matching the Perspectives of User & Document Profiles for Effective Retrieval.

Researcher

Mr. J. Hurlock

Objective of Study

The study wishes to gather a large amount of data regarding weights for ranking on a search engine. The weightings relate to certain criteria based on the following paper 'Searching Twitter: Separating the Tweet from the Chaff.' By Hurlock & Wilson.

We do this by, firstly performing searches at timed intervals, and retrieving the results, then asking participants to rate individual results for the given query on a scale, which then allows us to calculate DCG (Discounted Cumulative Gain), which will allow us to add weighted measures to each of the criteria we found in our initial study.

In the sections below I will describe parts of the study in greater detail.

Participants

We wish to recruit participants via the crowd sourcing platform CrowdFlower. CrowdFlower is the world's leading crowdsourcing service, with over 800 million tasks submitted by over four million contributors. They specialize in microtasking: distributing small, discrete tasks to many online contributors, assembly-line fashion - for instance, using people to check hundreds of thousands of photos every day for obscene content.

The number of participants we wish to use is based on the data we gather. However, for each query and each result we need a minimum of three participants to acquire a meaningful DCG score.

Participants will be compensated based on the number judgments they complete and amount of data gathered by the initial data collector. The exact amount will be calculated once all data is collected. However we will be basing the payment rated based on works by Ducan Watts and Winter Mason (see 'Financial incentives and the "performance of crowds"' ACM SIGKDD) Though other authors have looked at the financial incentives (Horton, John Joseph, and Lydia B. Chilton. "The labor economics of paid crowdsourcing." Proceedings of the 11th ACM conference on Electronic commerce. ACM, 2010. , Paolacci, Gabriele, Jesse Chandler, and Panagiotis Ipeirotis. "Running experiments on amazon mechanical turk." Judgment and Decision Making 5.5 (2010): 411-419., Buhrmester, Michael, Tracy Kwang, and Samuel D. Gosling. "Amazon's Mechanical Turk A New Source of Inexpensive, Yet High-Quality, Data?." Perspectives on Psychological Science 6.1 (2011): 3-5.)

CrowdFlower also allows for gold standards to be put into the data, to check that participants are not bots. Participants will be scored for accuracy. If they fail these gold standard test, then they will be considered a bot and will not receive any compensation. More information about gold data can be found at http://crowdflower.com/docs/gold

Data

There are two types of data involved with this study firstly the query data, and then the returned data. We wish to query the Twitter search engine for 24 hours at each hour, for a set of queries. This will then return a list of results. In the sections below I will describe the data in more detail.

Query Data

Just as you type query data into search engines such as Google, you can do the same in Twitter's search engine. The query data we are using are slightly modified queries based on three information retrieval tasks found in our initial study, one in a location based query, one is a temporal based query and the other is an information specific query. For instance in the first study we asked people to find somewhere to eat lunch in London, we asked people to find information about the BBC proms (temporal), and finally we asked them imagine they wanted to buy a new iPhone and to find information about it.

The queries that will be submitted to the search engine will be based on a query log submitted in our first study, modified so that the event (BBC Proms) and device (iPhone) will be more relevant to today.

Returned Data

When the queries are submitted to the search engine, results are returned, in the form of tweets. These will be used with the query string and a time stamp to ask the user for a rating of usefulness.

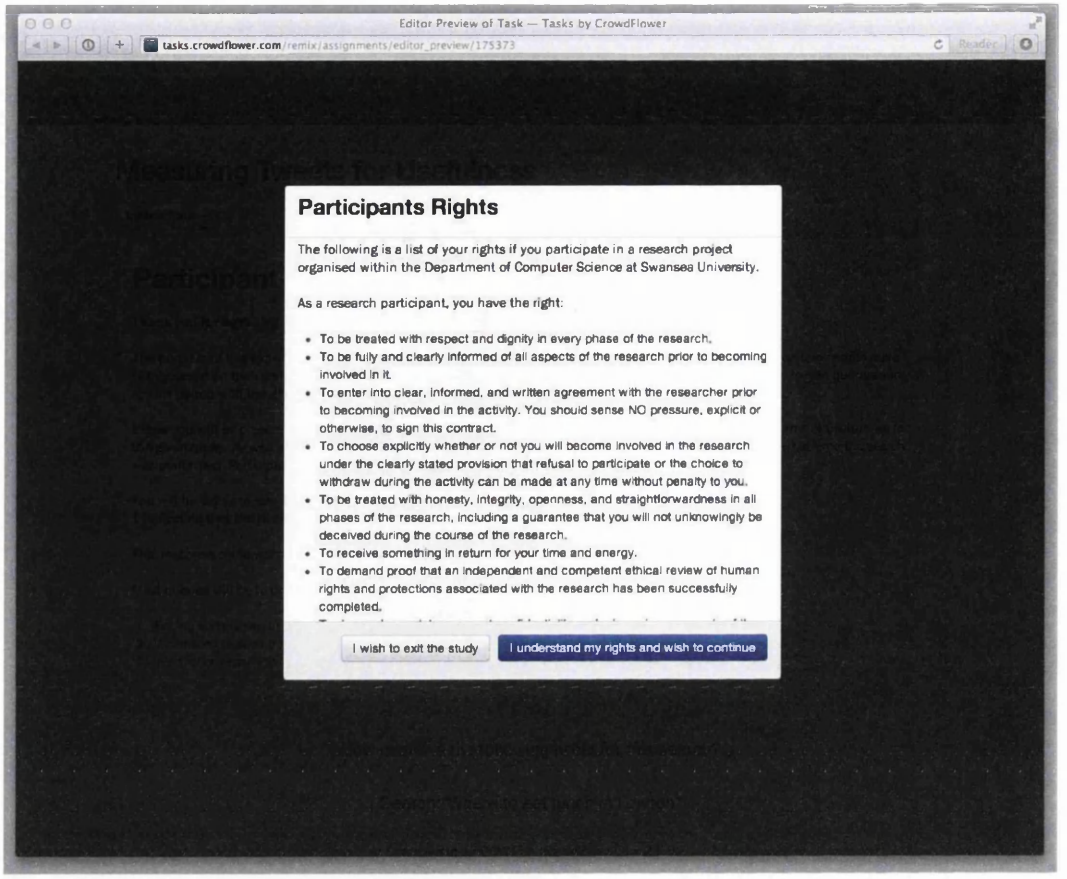
As this data is from a public data source, there may be offensive content in the tweet itself and/or the links contained in the tweets. Participants may find offensive (e.g. swear words) We wil attempt to remove all such material however, there is a very small likelihood of someone still being offended. To further mitigate this participants are warned of this at the top of the page and are told they may be offended by the content in the tweet, or any external website the tweet links to. They are explicitly told they may leave the study at any time.

Example Interface & Instructions for Participants

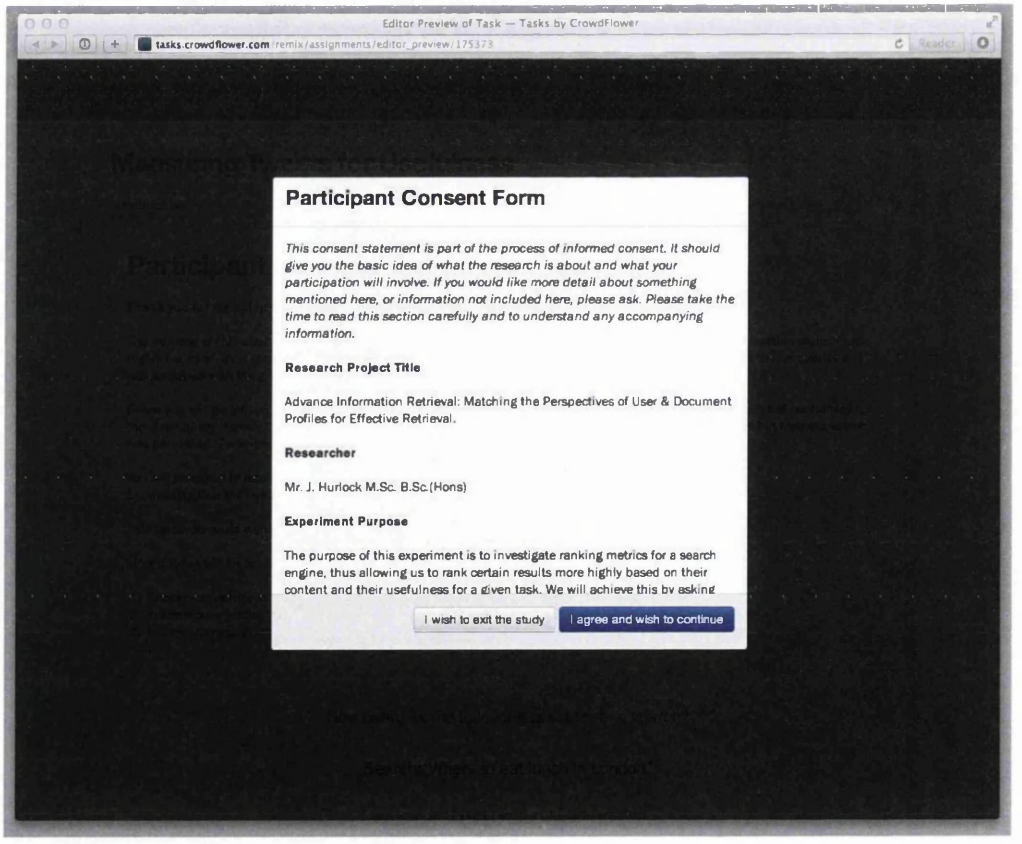
An example of instructions, consent form, participants rights, as well as the interface presented to the participants can be found at the following URL:

Long URL :
https://tasks.crowdflower.com/remix/assignments/editor_preview/175373
Short URL: http://bit.ly/Yh5NBv

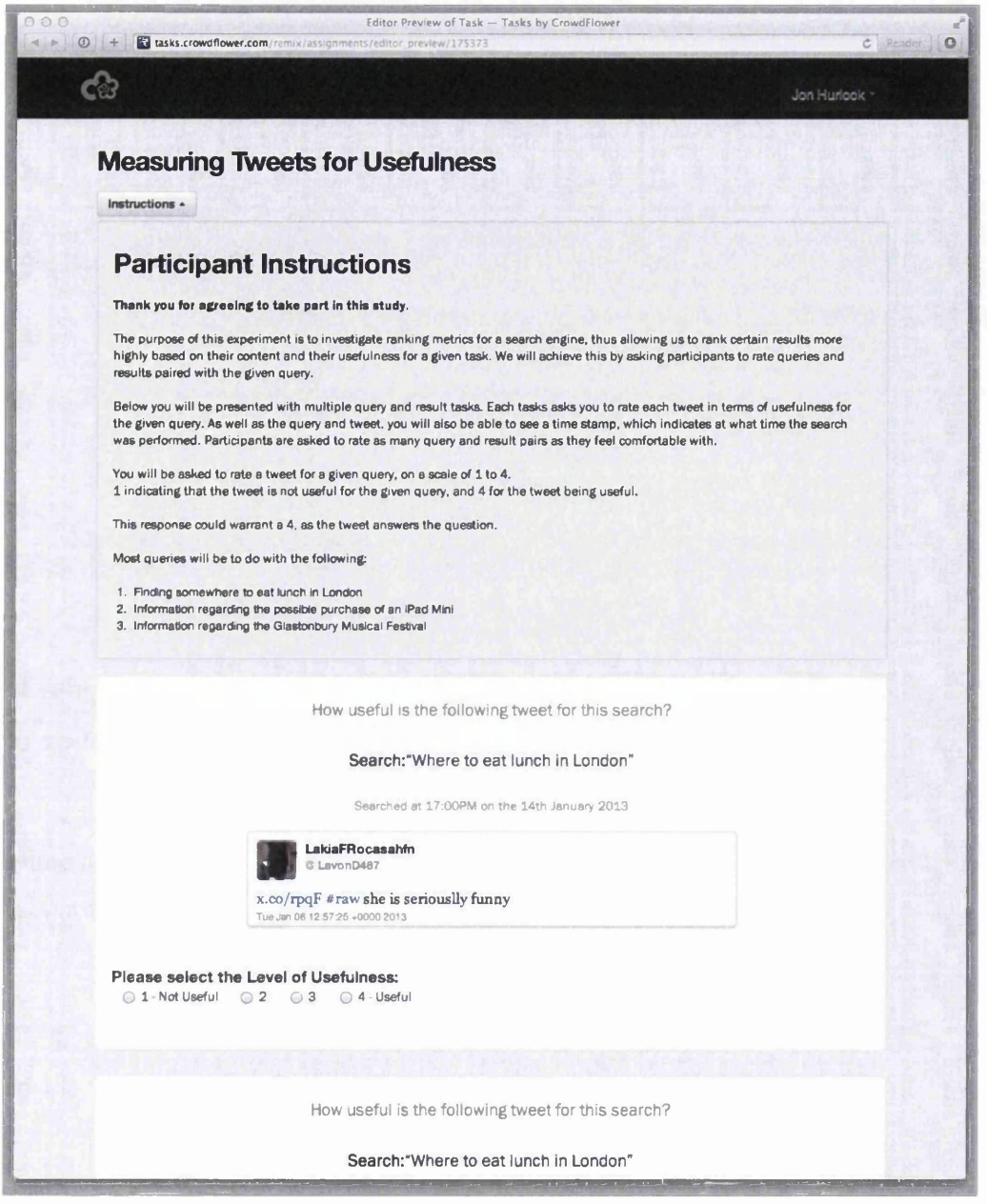
Screenshots of the user agreement process is shown on the following pages, as well as the interface. If the user does not consent or agree to either the participant rights or the consent form, then the can press the I wish to exit the study button, which will redirect them to Google.com



Screen shot showing participants rights form, which appears when page is loaded.



Screen shot showing participants consent form, which appears once the user agrees to participants rights form.



A page showing the instructions and one of the tasks

Swansea University – Computer Science Department

Research Participant's Bill of Rights

The following is a list of your rights if you participate in a research project organised within the Department of Computer Science at Swansea University.

As a research participant, you have the right:

- To be treated with respect and dignity in every phase of the research.
- To be fully and clearly informed of all aspects of the research prior to becoming involved in it.
- To enter into clear, informed, and written agreement with the researcher prior to becoming involved in the activity. You should sense NO pressure, explicit or otherwise, to sign this contract.
- To choose explicitly whether or not you will become involved in the research under the clearly stated provision that refusal to participate or the choice to withdraw during the activity can be made at any time without penalty to you.
- To be treated with honesty, integrity, openness, and straightforwardness in all phases of the research, including a guarantee that you will not unknowingly be deceived during the course of the research.
- To receive something in return for your time and energy.
- To demand proof that an independent and competent ethical review of human rights and protections associated with the research has been successfully completed.
- To demand complete personal confidentiality and privacy in any reports of the research unless you have explicitly negotiated otherwise.
- To expect that your personal welfare is protected and promoted in all phases of the research, including knowing that no harm will come to you.

- To be informed of the results of the research study in a language you understand.
- To be offered a range of research studies or experiences from which to select, if the research is part of fulfilling your educational or employment goals.

The contents of this bill were prepared by the University of Calgary who examined all of the relevant Ethical Standards from the Canadian Psychological Association's Code of Ethics for Psychologists, 1991 and rewrote these to be of relevance to research participants.

Descriptions of the CPA Ethical Code and the CPA Ethical Standards relevant to each of these rights are available at http://www.cpa.ca/ethics2000.html and http://www.psych.ucalgary.ca/Research/ethics/bill/billcode.html if you would like to examine them.

The complete CPA Ethical Code can be found in Canadian Psychological Association *"Companion manual for the Canadian Code of Ethics for Psychologists"* (1992).

Swansea University – Computer Science Department

Research Consent Form

This consent statement is part of the process of informed consent. It should give you the basic idea of what the research is about and what your participation will involve. If you would like more detail about something mentioned here, or information not included here, please ask. Please take the time to read this section carefully and to understand any accompanying information.

Research Project Title

Advance Information Retrieval: Matching the Perspectives of User & Document Profiles for Effective Retrieval.

Researcher

Mr. J. Hurlock

Experiment Purpose

The purpose of this experiment is to investigate ranking metrics for a search engine, thus allowing us to rank certain results more highly based on their content and their usefulness for a given task. We will achieve this by asking participants to rate queries and results paired with the given query.

Participant Recruitment and Selection

Participants are recruited through crowdflower, a crowd sourcing engine. Participants will only be compensated for their time if they complete the tasks to a satisfactory level (see: http://crowdflower.com/solutions/self-service/faqs)

Procedure

Below you will be presented with multiple query and result tasks. Each task asks you to rate each tweet in terms of usefulness for the given query. As well as the query and tweet, you will also be able to see a time stamp, which indicates at what time the search was performed. All links on the tweets are 'live' and can be clicked by participants. Participants are asked to rate as many query and result pairs as they feel comfortable with.

Data Collection

Crowdflower collects data about each participant regarding their location, time they completed task, as well as the service used to access the crowdflower platform. Most importantly crowdflower collects data regarding the participants actions during the task such as which answers were given for each task, as well as a trust metric, to help filter out bots and users misusing the service.

Data Archiving/Destruction

Data will be kept securely. The investigator will destroy study data after it is no longer of use. Usually, this will be at the end of the research project when results are fully reported and disseminated.

Confidentiality

Confidentiality and participant anonymity will be strictly maintained. All information gathered will be used for statistical analysis only and no names or other identifying characteristics will be stated in the final or any other reports.

Likelihood of Discomfort

We have gathered data from a publically available dataset (Twitter Search API), some content in the dataset may be explicit/offensive in nature. While every effort has been made to avoid exposing participants to offensive/explicit content, we cannot always guarantee that you will not be exposed to such content.

We have attempted to remove and filter out any content or links that may contain offensive/explicit content. However, you may find that some of the content from external sites/links has changed since we reviewed the content.

In the event you find some of the content offensive or discomforting, you may quit the study at any time or alternatively move onto the next task, if you feel comfortable to do

so. If you want to report any offensive / explicit content to us, you may do so by emailing the researcher using the email address found in the next section.

Researcher

Mr. J Hurlock is working on his doctorate in the Computer Science Department at the Swansea University. This study will contribute to his research regarding advanced information retrieval. His supervisor is Prof Matt Jones.

J Hurlock can be contacted in room 500 Faraday Tower, Swansea University, Singleton Park, Swansea, SA2 8PP, United Kingdom. His email address is j[dot]Hurlock[at]swan[dot]ac[dot]uk

Finding out about Results

Participants can find out the results of the study by contacting the researcher after February 1, 2014.

Agreement

By taking part in this study you have understood to your satisfaction the information regarding participation in the research project and agree to take part as a participant. In no way does this waive you legal rights nor release the investigators, sponsors, or involved institutions from their legal and professional responsibilities. You are free to not answer specific items or questions in interviews or on questionnaires. You are free to withdraw from the study at any time without penalty. Your continued participation should be as informed as your initial consent, so you should feel free to ask for clarification or new information throughout your participation. If you have further questions concerning matters related to this research, please contact the researcher.

To: **Matt Jones**, Ethics Committee Chair

From: **J Hurlock**

Date: March 12th, 2013

Re: **Request for experiment approval**

This memo is a request for approval to perform several experiments involving human participants via the crowd sourcing platform crowdflower during the months April this year. All the experiments will take place via crowdflower's crowd sourcing engine (http://www.crowdflower.com).

I have read and understood the Swansea CS Ethics Regulations. This application includes all relevant information necessary for evaluation against these regulations.

Attached to this memo you will find a Research Participant's Bill of Rights and a Research Consent Form. A copy of the Bill of Rights will be displayed to the participant before they engage in the tasks. A copy of the Research Consent Form will be also be displayed to the participant before any tasks.

At the beginning of each page instructions at the top of the page will explain these documents, with particular reference to the participant's right to withdraw at any point without explanation, the participant will then have a choice to proceed with the study or withdraw. The participant is also warned that the content is sourced from a public data stream, and that some content may be regarded as offensive, and if they wish to with draw from the study at some point they may do so. They are also told that we can not be held responsible for any content hosted on external websites found via links in the data.

A copy of the participant's written instructions are also attached to this memo. The instructions include an introduction and a brief explanation of the purpose of the experiment.

Participants will be asked to solve as many tasks as feel comfortable with. Some tasks are included to test the participant's accuracy, this is to prevent 'bots' and 'random clicks' from gaming the crowdflower engine.

If you require any further information, I can be found in room 500, or emailed at 323358@swan.ac.uk

Signed:

Supervisor's signature:

Date: March 2013

Bibliography

Alonso, O., & Baez-Yates, R. (2011). Design and implementation of relevance assessments using crowdsourcing. *In Advances in information retrieval* (pp. 153-164). Berlin: Springer.

Alonso, O., Marshall, C. C., & Najork, M. (2013). Are Some Tweets More Interesting Than Others? #HardQuestion. *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval .*

Andre, P., Bernstein, M. S., & Luther, K. (2012). Who Gives A Tweet? Evaluating Microblog Content Value. *In Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work* (pp. 471-474). Seattle: ACM.

Aramaki, E., Maskawa, S., & Morita, M. (2011). Twitter catches the flu: detecting influenza epidemics using Twitter. *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1168-1576). Stroudsburg,PA: Association for Computational Linguistics.

Bakshy, E., Hofma, J. M., Mason, W. A., & Watts, D. J. (2011). Everyone's an influencer: quantifying influence on twitter. *n Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 65-74). ACM.

Barnett, E. (2010). *Facebook hits 500m: social media by numbers.* From The Telegraph: http://www.telegraph.co.uk/technology/facebook/7903071/Facebook-hits-500m-social-media-by-numbers.html

Barry, C. L., & Schamber, L. (1998). Users' criteria for relevance evaluation: a cross-situational comparison. *Information processing & management , 34* (2), 219-236.

Beaumont, C. (2008). *Mumbai attacks: Twitter and Flickr used to break news.* From The Telegraph: http://www.telegraph.co.uk/news/worldnews/asia/india/3530640/Mumbai-attacks-Twitter-and-Flickr-used-to-break-news-Bombay-India.html

Beaumont, C. . *New York plane crash: Twitter breaks the news, again.* From The Telegraph: http://www.telegraph.co.uk/technology/twitter/4269765/New-York-plane-crash-Twitter-breaks-the-news-again.html

Beaumont, P. (2011). *The truth about Twitter, Facebook and the uprisings in the Arab world.* From The Guardian: http://www.theguardian.com/world/2011/feb/25/twitter-facebook-uprisings-arab-libya

Bernstein, M. S., Suh, B., Hong, L., Chen, J., Kairam, S., & Chi, E. H. (2010). Eddi: interactive topic-based browsing of social status streams. *Proceedings of the 23nd annual ACM symposium on User interface software and technology* (pp. 303-312). ACM.

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python - Analyzing Text with the Natural Language Toolkit*. Sebastopol, California, USA: O'Reilly Media.

Boolen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science , 2* (1), 1-8.

Boyd, D. M., & Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. 2007. *Journal of Computer-Mediated Communication , 13*, 210-230.

Boyd, D., Golder, S., & Lotan, G. (2010). Tweet, Tweet, Retweet: Conversational Aspects of Retweeting on Twitter. *HICSS-43 IEEE: Proceedings of the 2010 43rd Hawaii International Conference on System Sciences* (43).

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems, , 30* (1), 107-117.

British Broadcasting Corporation. (2013). *Twitter launches UK alert service for emergencies*. From BBC News: http://www.bbc.co.uk/news/technology-24986263

Broder, A. (2002). A taxonomy of web search. *ACM SIGIR Forum , 36* (2).

Buhrmester, M., Kwang, T., & Goslin, S. D. (2011). Amazon's Mechanical Turk a new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science , 6* (1), 3-5.

Busch, M., Gade, K., Larson, B., Lok, P., Luckenbill, S., & Lin, J. (2012). Earlybird: Real-Time Search at Twitter. *Data Engineering (ICDE), 2012 IEEE 28th International Conference on* (pp. 1360-1369). IEEE.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. K. (2010). Measuring User Influence in Twitter: The Million Follower Fallacy. *Proceedings of the Fourth AAAI International Conference on Weblogs and Social Media*. AAAI.

Cherichi, S., & Faiz, R. (2013). New metric measure for the improvement of search results in microblogs. *Proceedings of The 3rd International Conference on Web Intelligence, Mining and Semantics*. ACM.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement , 20* (1), 37-46.

Culotta, A. (2010). Towards Detecting Influenza Epidemics by Analyzing Twitter Messages. *Proceedings of the First Workshop on Social Media Analytics* (pp. 115-122). New York, NY: ACM.

Derczynski, L., Ritter, A., Clark, S., & Bontcheva, K. (2013). Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data. *Proceedings of the International Conference on Recent Advances in Natural Language Processing*. Association for Computational Linguistics.

Diakopoulos, N. A., & Shamma, D. A. (2010). Characterizing Debate Performance via Aggregated Twitter Sentiment. *Proceedings of the 28th international conference*

on Human factors in computing systems (ACM CHI '10) (pp. 1195-1198). Atlanta, Georgia: ACM.

Diakopoulos, N., Naaman, M., & Kivran-Swaine, F. (2010). Diamonds in the Rough: Social Media Visual Analytics for Journalistic Inquiry. *Proceedings of the IEEE Conference on Visual Analytics Science and Technology, IEEE VAST 2010*. Salt Lake City, Utah: IEEE.

Dorsey, J. (2011). Foundation 01 // Jack Dorsey. *Foundation*. (K. Rose, Interviewer) youtube.

Efron, M., & Winget, M. (2010). Questions are content: a taxonomy of questions in a microblogging environment. *ASIS&T 2010: Proceedings of the 73rd ASIS&T Annual Meeting on Navigating Streams in an Information Ecosystem. 47*. Silver Springs, MD: American Society for Information Science.

Elasticsearch. (2014). *query dsl » queries » function score query*. From Elastic Search Reference [1.x] : http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl-function-score-query.html#_using_function_score

Elastisearch. (2014). *analysis » analyzers » custom analyzer*. From Elasticsearch Reference [1.x]: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/analysis-custom-analyzer.html

Elsweiler, D., & Harvey, M. (2014). Engaging and maintaining a sense of being informed: Understanding the tasks motivating twitter search. *Journal of the American Society for Information Science and Technology* .

Elsweiler, D., Wilson, M. L., & Kirkegaard Lunn, B. (2011). Understanding casual-leisure information behaviour. *Library and Information Science , 1*, 211-241.

Evans, B. M., & Chi, E. H. (2008). Towards a model of understanding social search. *Proceedings of the 2008 ACM conference on Computer supported cooperative work* (pp. 485-495). ACM.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin , 76* (5), 378-382 .

Fogg, B. J., Marshall, J., Laraki, O., Osipovich, A., vARMA, C., Fang, N., et al. (2001). What makes Web sites creible?: a report on a large quantiative study. *In Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 61-68). ACM.

Gaffney, D. (2010). #iranElection: Quantifying Online Activism. *Proceedings of the WebSci10: Extending the Frontiers of Society On-Line*. Raleigh, NC, USA: Web Science Trust.

Glaser, B. G., & Strauss, A. L. (2009). *The Discovery of Grounded Theory: strategies for qualitative research*. Piscataway, New Jersey, USA: Transaction Publishers.

Grossman, L. (2009). *Iran Protests: Twitter, the Medium of the Movement*. From Time: http://content.time.com/time/world/article/0,8599,1905125,00.html

Hansell, S. (2007). *Google Keeps Tweaking Its Search Engine*. Retrieved 2014 28-August from New York Times: http://www.nytimes.com/2007/06/03/business/yourmoney/03google.html?pagewanted=all&_r=0

Harris, J. K., & Burke, R. C. (2005). Do you see what I see? An application of inter-coder reliability in qualitative analysis. *American Public Health Association 133rd Annual Meeting & Exposition*. Washington, DC, USA: American Public Health Association.

Hawn, C. (2009). Take two aspirin and tweet me in the morning: how Twitter, Facebook, and other social media are reshaping health care. *Health Affairs* , 361-368.

Horton, J. J., & Chilton, L. B. (2010). The labor economics of paid crowdsourcing. *In Proceedings of the 11th ACM conference on Electronic commerce* (pp. 209-218). ACM.

Hurlock, J. (2010). *Searching Twitter: Extracting Useful Information*. Swansea, Wales: Swansea University.

Hurlock, J., & Wilson, M. L. (2011). Searching Twitter: Separating the Tweet from the Chaff. *The Fifth International AAAI Conference on Weblogs and Social Media (ICWSM)*. Barcelona: Association for the Advancement of Artificial Intelligence.

Imran, M., Elbassuoni, S. M., Castillo, C., Diaz, F., & Meier, P. (2013). Extracting information nuggets from disaster-related messages in social media. *In Proceedings of of the 10th International Conference on Information Systems for Crisis Response and Management*. ISCRAM.

Jabeur, L. B., Tamine, L., & Boughanem, M. (2012). Uprising Microblogs: A Bayesian Network Retrieval Model for Tweet Search. *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (pp. 943-948). ACM.

Jansen, B. J., Booth, D. L., & Spink, A. (2007). Determining the informational, navigational, and transactional intent of Web queries. *Information Processing and Mnagaement* , 1251-1266.

Jansen, B. J., Spink, A., & Saracevic, T. (2000). Real life, real users, and real needs: a study and analysis of user queries on the web. *Information processing & management* , 207-227.

Java, A., Song, X., Finin, T., & Tseng, B. (2007). Why we twitter: understanding microblogging usage and communities. *WebKDD/SNA-KDD 2007: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*. New York, NY: ACM.

Jeffrey, D., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM , 51* (1), pp. 107-113.

Karttunen, L. (1977). Syntax and semantics of questions. *Linguistics and Philosophy* .

Krikorian, R. (2010). *Map of a Twitter Status Object*. From Wall Street Journal: http://online.wsj.com/public/resources/documents/TweetMetadata.pdf

Krippendorff, K. (2013). *Computing Krippendorff's Alpha-Reliability*. From University of Pennsylvania Klaus Krippendorff's Home Page: http://www.asc.upenn.edu/usr/krippendorff/mwebreliability5.pdf

Kulkami, A., Teevan, J., Svore, K. M., & Dumais, S. T. (2011). Understanding temporal query dynamics. *In Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 167-176). ACM.

Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is Twitter, a socila network or a news media? *Proceedings of the 19th international conference on World wide web* (pp. 591-600). ACM.

Lampos, V., De Bie, T., & Cristianini, N. (2010). Flu detector-tracking epidemics on Twitter. In J. Balcazar, F. Bonchi, A. Gionis, & M. Sebag, *Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science* (Vol. 6323, pp. 599-602). Berlin, Heidelberg: Springer.

Landis, R. J., & Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics , 33* (1), 159-174.

Lease, M., & Yilmaz, E. (2012). Crowdsourcing for information retrieval. In *ACM SIGIR Forum* (Vol. 45). ACM.

Likert, R. (1932). *A technique for the measurement of attitudes*. Archives of Psychology.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Manning, C., & Jurafsky, D. (2015, August 3). *Stanford Natural Language Processing - Week 7 - Ranked Information Retrieval*. Retrieved August 3, 2015, from Coursera: https://class.coursera.org/nlp/lecture

McCreadie, R., & MacDonald, C. (2013). Relevance in microblogs: enhancing tweet retrieval using hyperlinked documents. *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

Morris, M. R., Teevan, J., & Panovich, K. (2010). What Do People Ask Their Social Networks and Why? A Survey Study of Status Message Q&A Behaviour. *CHI 2010 Proceedings of the 28th international conference on Human factors in computing systems* (pp. 1739-1748). New York, NY: ACM.

Murphy, K. P. (2012). *Machine Learning A Probabilistic Perspective*. Cambridge, MA, USA: The MIT Press.

Naaman, M., Boase, J., & Lai, C. (2010). Is it really about me? Message Content in Social Awareness Streams. *In Proceedings of the 2010 ACM conference on Computer supported cooperative work* (pp. 189-192). ACM.

Naveed, N., Gottron, T., Kunegis, J., & Alhadi, A. C. (2011). Bad news travel fast: A content-based analysis of interestingness on twitter. *Proceedings of the 3rd International Web Science Conference*. ACM.

Neumeyer, L., Robbins, B., Nair, A., & Kesari, A. (2010). S4: Distributed stream computing platform. *In Data Mining Workshops (ICDMW), 2010 IEEE International Conference on* (pp. 170-177). IEEE.

Noll, M. G. (2007). *Writeing an Hadoop MapReduce Program in Python*. Retrieved 2014 7-March from Micheal G. Noll. Applied Research. Big Data. Distributed Systems. Open Source.: http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/

Noyes, K. (2014). *How a little open source project came to dominate big data*. From Fortune: http://fortune.com/2014/06/30/hadoop-how-open-source-project-dominate-big-data/

Ott, M., Cardie, C., & Hancock, J. T. (2013). Negative Deceptive Opinion Spam. *Proceedings of NAACL-HLT 2013* (pp. 497-501). Atlanta Georgia: Association for Computational Linguistics.

Ounis, I., MacDonald, C., Lin, J., & Soboroff, I. (2011). The Twentieth Text REtrieval Conference (TREC 2011) Proceedings. *In Proceedings of the Twentieth Text REtrieval Conference (TREC 2011)*. NIST.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., & Smith, N. A. (2013). Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters. *Proceedings of NAACL-HLT* (pp. 380-390). Atlanta: Association for Computational Linguistics.

Pal, A., & Counts, S. (2011). Identifying topical authorities in microblogs. *Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 45-54). ACM.

Pear Analytics. (2009). *Twitter Study - August 2009*. Retrieved 2014-January from Pear Analytics: http://www.pearanalytics.com/wp-content/uploads/2012/12/Twitter-Study-August-2009.pdf

Petrovic, S., Osborne, M., & Lavrenko, V. (2010). Streaming first story detection with application to twitter. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 181-189). ACL.

Petrovic, S., Osborne, M., & Lavrenko, V. (2010). The Edinburgh Twitter Corpus. *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*. (pp. 25-26). NAACL.

Phuvipadawat, S., & Murata, T. (2010). Breaking news detection and tracking in twitter. *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on. 3*, pp. 120-123. IEEE.

Qu, Y., Huang, C., Zhang, P., & Zhang, J. (2011). Microblogging after a major disaster in China: a case study of the 2010 Yushu earthquak. *Proceedings of the ACM 2011 conference on Computer supported cooperative work* (pp. 25-34). ACM.

Rello, L., & Baeza-Yates, R. A. (2012). Social Media Is NOT that Bad! The Lexical Quality of Social Media. *ICWSM*. AAAI.

Ritterman, J., Osborne, M., & Klein, E. (2009). Using prediction markets and Twitter to predict a swine flu pandemic. *In 1st international workshop on mining social media, 9.*

Sanders Analytics. (2011). *Twitter Sentiment Corpus*. Retrieved August 10, 2015, from Sanders Analytics: http://www.sananalytics.com/lab/twitter-sentiment/

Schneiderman, B., Byrd, D., & Croft, B. W. (1997). Clarifying search: A user-interface framework for text searches. In A. Friedlander (Ed.), *D-lib Magazine: The Magazine of Digital Library Research*. Corporation for National Research Initiatives.

Search Engine Land. (2011). *As Deal With Twitter Expires, Google Realtime Search Goes Offline*. Retrieved 2014 24-January from Search Engine Land: http://searchengineland.com/as-deal-with-twitter-expires-google-realtime-search-goes-offline-84175

Search Engine Land. (2013) *Bing, Twitter Renew Deal To Include Tweets In Search Results*. Retrieved 2014 24-January from Search Engine Land: http://searchengineland.com/bing-twitter-renew-deal-to-include-tweets-in-search-results-175791

Segaran, T. (2007). *Programming Collective Intelligence - Building Smart Web 2.0 Applications*. Sebastopol, CA, USA: O'Reilly Media.

Shamma, D., Kennedy, L., & Churchill, E. Tweetgeist: Can the twitter timeline reveal the structure of broadcast events. *CSCW Horizons*. ACM.

Sharoda, A. P., Hong, L., & Chi, E. H. (2011). Is Twitter a Good Place for Asking Questions? A Characterization Study. *Proceedings of the Fifth AAAI International Conference on Weblogs and Social Media*. AAAI.

Soboroff, I., McCullough, D., Lin, J., MacDonald, C., Ounis, I., & McCreadie, R. (2012). Evaluating real-Time Search over Tweets. *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*. AAAI.

Spärck-Jones, K., & van Rijshergen, C. J. (1975). *Report on the need for and provision of an 'ideal' information retrieval test collection*. Computer Laboratory, University of Cambridge, BL R&D Report 5266. British Library.

Spink, A., Greisdorf, H., & Bateman, J. (1998). From highly relevant to not relevant: examining different regions of relevance. *Information Processing & Management , 34* (5), 599-621.

Sullivan, D. (2009). *Up Close with Bing's Twitter Search Engine*. From Search Engine Land: http://searchengineland.com/live-today-bings-twitter-search-engine-28224

Talbot, D. (2010). *How Google Ranks TWEETS*. From MIT Technology Review: http://www.technologyreview.com/news/417085/how-google-ranks-tweets/

Teevan, J., Ramage, D., & Ringel Morris, M. #TwitterSearch: a comparison of microblog search and web search. *Proceedings of the fourth ACM international conference on Web search and data mining (WSDM)*. Hong Kong: Association for Computing Machinery.

Teufel, S. (2006). *Information Retrieval Lecture 3: Evaluation methodology*. Retrieved August 5, 2015, from University of Cambridge, Computer Laboratory: Information Retrieval 2006-07: https://www.cl.cam.ac.uk/teaching/2006/InfoRtrv/lec3.2.pdf

The Apache Software Foundation. (2013). *Hadoop 1.1.2 Documentation - MapReduce - Hadoop Streaming*. Retrieved 2014 7-March from Apache.org: http://hadoop.apache.org/docs/r1.1.2/streaming.html#Hadoop+Streaming

Thorsten, J. (2006). Training linear SVMs in linear time. *n Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 217-226). ACM.

Tsur, O., Littman, A., & Rappoport, A. (2013). Efficient Clustering of Short Messages into General Domains. *International AAAI Conference on Weblogs and Social Media; Seventh International AAAI Conference on Weblogs and Social Media (ICWSM)*. AAAI.

Tunkelang, D. (2009). *A Twitter Analog to PageRank*. From The Noisy Channel: http://thenoisychannel.com/2009/01/13/a-twitter-analog-to-pagerank

Twitter Inc. (n.d.). *FAQs about Retweets (RT)*. From Twitter Help Center: https://support.twitter.com/articles/77606-faqs-about-retweets-rt

Twitter Inc. (2008). *Finding A Perfect Match*. From Twitter Blog: https://blog.twitter.com/2008/finding-perfect-match

Twitter Inc. (2014). *On Twitter's 8th birthday, find your first Tweet*. From Twitter Blog: https://blog.twitter.com/2014/on-twitters-8th-birthday-find-your-first-tweet

Twitter Inc. (n.d.). *Tweets*. From Twitter Developers: https://dev.twitter.com/overview/api/tweets

Twitter, Inc. (2011). *200 million Tweets per day*. From Twitter Blog: https://blog.twitter.com/2011/200-million-tweets-day

Twitter, Inc. (2013). *Celebrating #Twitter7*. Retrieved 2014 21-January from Twitter Blogs: https://blog.twitter.com/2013/celebrating-twitter7

Twitter, Inc. (2010). *Measuring Tweets*. Retrieved 21 2014-January from Twitter Blogs: https://blog.twitter.com/2010/measuring-tweets

Vieweg, S., Hughes, A. L., Starbird, K., & Palen, L. (2010). Microblogging during two natural hazards events: what twitter may contribute to situational awareness. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1079-1088). ACM.

Wang, A. H. (2010). Don't follow me: Spam detection in twitter. *In Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on* (pp. 1-10). IEEE.

Welinder, P., & Pietro, P. (2010). Online crowdsourcing: rating annotators and obtaining cost-effective labels. *In Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 25-32). IEEE.

Weng, J., Lim, E. P., Jiang, J., & He, Q. (2010). Twitterrank: finding topic-sensitive influential twitterers. *In Proceedings of the third ACM international conference on Web search and data mining* (pp. 261-270). ACM.

Wikipedia. (2015, Aug 17). *Brown Corpus*. Retrieved Aug 17, 2015, from Wikipedia: https://en.wikipedia.org/wiki/Brown_Corpus

Williams, E. (2010). *A Better Twitter*. From Twitter Blog: http://blog.twitter.com/2010/09/better-twitter.html

Wilson, M. L., & Elsweiler, D. (2010). Casual-leisure Searching: the Exploratory Search scenarios that break our current models. *HCIR '10: 4th International Workshop on Human-Computer Interaction and Information Retrieval*. New York, NY, USA: ACM.

Winter, M., & Watts, D. J. (2010). Financial incentives and the performance of crowds. *ACM SIGKDD Explorations Newsletter* (pp. 100-108). ACM.

Yang, J., & Leskovec, J. (2011). Patterns of Temporal Variation in Online Media. *WSDM'11*. Hong Kong, China: ACM.

Zhao, D., & Rosson, M. B. (2009). How and Why People Twitter: The Role that Micro-blogging plays in informal communication at work. *In Proceedings of the ACM 2009 international conference on Supporting group work* (pp. 243-252). ACM.

Zuckerberg, M. (2010). *500 Million Stories*. From Facebook: https://www.facebook.com/notes/facebook/500-million-stories/409753352130