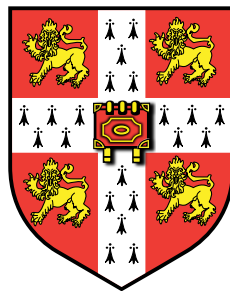


Measuring the mood of the world



Alex Davies

Department of Engineering

University of Cambridge

A first year report submitted for the degree of

Doctor of Philosophy

June 2011

Abstract

People are increasingly posting public content on the internet that reveals their sentiments and opinions. By far the largest source of this content, in terms of numbers of users, is Twitter. Twitter, at the time of writing, has 200 million users, who generate over 65 million tweets per day. There is a vast amount of sentiment information contained in this data, but we need suitable techniques to effectively extract it.

Understanding the sentiment or opinions of people is a valuable resource. It is useful on both a macro scale, where we can evaluate aggregated sentiment to predict macro-scale human behaviour, and a micro scale, where individual's sentiments provide actionable information to personalize services and target particular users.

We address the problem of language-independent sentiment modelling. We propose a probabilistic model that learns a word distribution for each sentiment based on a set of key indicator terms for each sentiment. Secondly, we propose an extension for modelling different sentiment distributions in different geographic regions, while incorporating information from neighbouring regions to improve estimates in areas of low tweet density.

Contents

Contents	ii
1 Literature Review	1
1.1 Features	2
1.1.1 Bag of Words	2
1.1.2 Bag of N-grams	3
1.1.3 Part of speech tags	4
1.1.4 Negation	5
1.2 Supervised Techniques	5
1.2.1 Naive Bayes	6
1.2.2 Support Vector Machines	7
1.2.3 Min Graph Cut	7
1.3 Semi-supervised learning/Unsupervised learning	8
2 Background	10
2.1 Labeled data	10
2.2 Key traits of Twitter	11
2.2.1 Language diversity	12
2.2.2 Structure of tweets	13
2.2.3 Spam	13
2.3 Predictive applications	14
3 A sentiment analysis method for Twitter	16
3.1 Motivation	16
3.1.1 Mixture models	16

3.1.2	Naive Bayes vs Mixture of Multinomials	17
3.1.3	Bayesian mixture of multinomials	18
3.2	Sentiment prediction	18
3.2.1	Likelihood	20
3.2.2	Prior	20
3.2.3	Learning	23
3.2.3.1	Variational approximation	23
3.2.3.2	Mean-field	24
3.2.4	Prediction	27
4	Spatial priors	28
4.1	Motivation	28
4.2	Region hierarchy	30
4.3	Prior	30
5	Experiments	34
5.1	Sentiment model comparison	36
5.2	Effect of incorporating geo-information	37
6	Future plan for Ph.D.	39
A	System implementation	42
A.1	Obtaining data	42
A.2	Filtering Tweets	43
A.3	Tweet Processing	44
A.4	Determining location in geo-hierarchy	45
A.5	Plotting	45
	Bibliography	47

Chapter 1

Literature Review

Sentiment analysis, or opinion mining, is a popular sub-field of natural language processing. It is concerned with analysing natural language texts, such as reviews, articles or comments, to determine the sentiment or feelings of the author, either over the whole text, or towards particular entities they refer to in the text. We will be concerned with the former case, where we are trying to determine the overall sentiment of a document. Specifically, given a document D , which is an ordered collection of words w , want to predict its sentiment s , from a fixed set of pre-chosen sentiments S .

Sentiment analysis emerged abruptly as a field around 2001, spurred on by a combination of improved techniques in machine learning and natural language processing and an increase in the availability of subjective texts, such as online reviews and blogs (Pang and Lee, 2008). Its status as an important field was reinforced by a growing realisation of the commercial applications of such technology. Automatically identifying consumer sentiment towards products and brands was identified early on as very useful application areas for business.

Sentiment analysis, like all fields of natural language processing, has to deal with the inherent complexity of natural language. While in many circumstances sentiment is expressed through simple constructions, eg “This product is terrible”, the full breadth of human ability to express sentiment is staggering. Use of literary devices such as simile, metaphor and sarcasm require a high level

“If you are reading this because it is your darling fragrance, please wear it at home exclusively, and tape the windows shut.”
“She runs the gamut of emotions from A to B.”
“This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can’t hold up.”

Table 1.1: List of non-trivial phrases for sentiment analysis from (Pang and Lee, 2008)

understanding of language and its meaning that is currently far beyond NLP techniques. Some examples of non-trivial sentences are shown in Table 1.1.

Even seemingly simple constructions can cause difficulty depending on document representation. For example: “I completely disagree that this product is bad”, contains the phrase “This product is bad” yet the overall sentiment is exactly the opposite. Fortunately, the domains to which these techniques are often applied contain texts that tend to be simpler.

1.1 Features

One of the most important parts of a NLP algorithm is choosing what features to use to represent a document. A full text string representation is not very useful, as there is no easy method for determining the conceptual similarity of two document-length strings. Instead we concentrate on simpler features.

1.1.1 Bag of Words

One of the simplest and most common string representations is the “Bag of Words” model. This ignores the ordering of words, and thus the structure of the sentence, and represents the document only as counts of the number of occurrences of the words in the document. An example is shown in Figure 1.1(a). Consequently, all permutations of a document are represented by the same vector. This obviously leads to the loss of fine-grained detail in sentences, as in the

and	1
flitting	1
is	2
never	1
raven	1
sitting	2
still	2
the	1

(a) Bag of words

< begin > and	1
and the	1
the raven	1
raven never	1
never flitting	1
flitting still	1
still is	2
is sitting	2
sitting still	1
sitting < end >	1

(b) Bag of bi-grams

Figure 1.1: Different representations of the sentence “And the raven never flitting, still is sitting, still is sitting”

case of “I don’t like him, I like her.” and “I don’t like her, I like him”.

Despite this obvious loss of information, the bag of words representation is still prevalent in a large array of applications, and performs very strongly. It is very computationally simple and in many applications much of the information required for learning is captured by this representation.

1.1.2 Bag of N-grams

A logical extension to the Bag of Words model is Bag of N-grams. In this representation, rather than storing counts of the occurrences of individual words, we store counts for groups of consecutive words of size n. This is illustrated in Figure 1.1(b).

This eliminates important ambiguities that are seen in bag of words models, such as “white house” being substantially different to “white horse was outside the house”.

There is a trade-off inherent in increasing the length of the grams in our representation. Obviously the advantage of increasing the length is that more context is captured by the representation. However, as the length is increased, the dimensionality of the input space increases exponentially, while the number of

non-zero elements for a document remains the same. Because of this, documents share far fewer common terms and all the documents become a similar distance apart.

Bigrams (2-grams) are very commonly used; in a lot of cases, the increased model variance that comes from the increase in sparsity is outweighed by the improved understanding of context that bigrams introduce. In some limited cases, trigrams (3-grams) are used, usually in the case where there is a very large training dataset that can sufficiently fill out the sparse representation. 4-grams and above are rarely used, and with large corpora will have a dimensionality of up to 10^{17} (Manning and Schütze, 1999).

Techniques exist that try and incorporate the benefits of different length n-grams. These usually employ methods to select a combination of different length n-grams, or to blend estimates of different length n-gram models (Katz, 1987).

1.1.3 Part of speech tags

A feature that has been found to be consistently useful in sentiment analysis (Pang and Lee, 2008) is POS-tagging (Part of speech tagging). A POS-tag is the grammatical category that a word belongs to in the context of a given sentence, ie whether a particular word is a noun, verb, adjective etc. In 1.2 is an example of a sentence that has been run through a POS-tagger.

State of the art POS taggers such as (Toutanova et al., 2003) use more advanced probabilistic models. Because many words can function as different parts of speech, we can not simply look up in a dictionary the set part-of-speech. In many cases, the problem can be ill-defined, as there may be inherent ambiguities as to a word's part-of-speech in a particular sentence. In this case, without additional context information, there is no single correct answer.

However, in practice modern POS-taggers have around a 96-97% accuracy rate, so we can consider this a very well solved problem. When used in sentiment

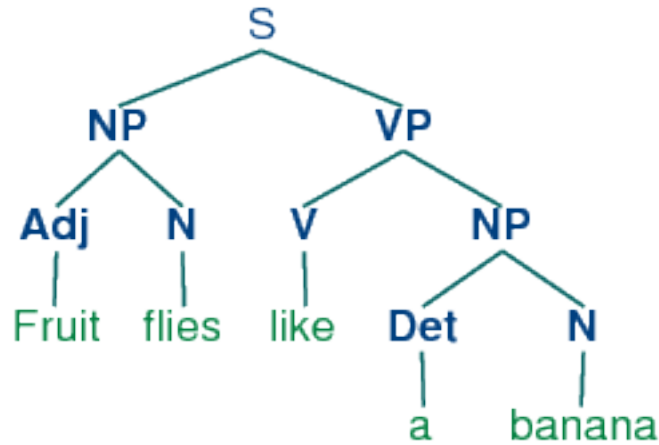


Figure 1.2: Parse tree for the sentence “Fruit flies like a banana” from (Bird et al.).

analysis applications, some methods find increases in performance, though there are conflicting results as to their utility (Pang et al., 2002) (Barbosa and Feng, 2010).

1.1.4 Negation

While there are many subtle structures that can fool basic sentiment analysis techniques, negation occurs so frequently that people tend to include it as special cases. This generally leads to increases in performance (Wilson et al., 2009) and also in *perceived* performance. This is important in cases where people will be manually interacting with the result of the sentiment prediction algorithm. In these cases people often perceive the quality of the algorithm by the number of “easy” examples that the system gets wrong, regardless of its overall accuracy.

1.2 Supervised Techniques

Once we have decided on a representation of our document, we need a method for converting this into a sentiment prediction. Initially, and in the majority of subsequent works, the problem is cast as one of supervised learning. That is, we have a training set of documents D_l that have had their sentiment identified by

a human expert. We then use these examples to learn a model of how our documents map to the given sentiments and use this to predict the sentiment of new documents without labels. To do this, standard supervised learning techniques, such as Naive Bayes, Support Vector Machines and Conditional Random Fields are often used.

1.2.1 Naive Bayes

One of the most simple supervised learning techniques for predicting discrete outputs (such as sentiment) from discrete inputs (such as text) is Nave Bayes. Despite its simplicity, it has been found to be highly effective in many areas with large datasets, especially NLP tasks such as spam detection in email. In these cases it often outperforms much more powerful techniques, such as SVMs, at a fraction of the computational cost.

The defining feature of Naive Bayes is it assumes that all features are independent given the class label. This is a very strong assumption, which is not true for most applications. For example, in a review of a new mobile phone, the word “huge” is likely to be negative. However, in a review of a house, the word huge is likely to be positive. Thus the meaning of “huge” is obviously not independent from its context. In general we would not expect that the meaning of a word is independent from the context of the surrounding words in the sentence.

The probability that a document \bar{x} (which is a vector of words), has a sentiment y is given as follows:

$$p(y|\bar{x}) \propto p(\bar{x}|y)p(y) \tag{1.1}$$

$$= \prod_{x \in \bar{x}} p(x|y)p(y) \tag{1.2}$$

1.2.2 Support Vector Machines

Support Vector Machines (SVMs) are one of the most popular and effective tools for general purpose classification available today. They work by attempting to find a hyperplane that separates the data into separate classes, while maintaining as much distance between the hyperplane and the points as possible. Importantly they can efficiently do this in very high dimensional spaces, such as the ones we use for sentiment analysis.

In some papers (Barbosa and Feng, 2010) (Pang et al., 2002), SVMs have shown promise in sentiment analysis, leading to decrease in error rate of 1-2% over a Naive Bayes baseline of 20%. However, other sources (Pak and Paroubek, 2010) claim that there is little improvement to be had over other techniques.

In the face of conflicting results, it is worth noting that more so than other classifiers, SVMs can be very sensitive to the tuning of their hyperparameters. That is, with poor settings, performance will be substantially worse than other techniques, but with a good setting, performance should be comparable with state of the art. Unfortunately, this tuning can be difficult and no standard process exists to find good settings. This is in contrast to principled probabilistic models, where parameters can be optimized via gradient methods. Details of hyperparameters and even kernels used are often omitted from papers so a detailed investigation is not possible.

1.2.3 Min Graph Cut

There are also methods that attempt to incorporate further information about the relatedness of documents. One of these is the min graph cut algorithm. This takes as input the output of one of the supervised classification methods as well as a matrix of similarities D between all of the documents. A graph is then constructed in the following manner:

1. A node n_{s_i} is created for each sentiment.
2. A node n_{d_j} is created for each document.

-
3. The weight on the edge between a node n_{s_i} and n_{d_j} , is a weight representing how likely it is the document d_j belongs to sentiment s_i based on the output of the previous classifier. This can, but does not have to, be a probability.
 4. The documents are then connected to all other documents; the edge weights for these are defined by the similarity measure D .
 5. Perform a min graph cut algorithm (?) on the graph.
 6. The resulting partitioning will place the documents in a cluster connected to exactly one of the class labels.

1.3 Semi-supervised learning/Unsupervised learning

There has also been substantial work looking at sentiment classification outside of the purely supervised classification setting. This is driven by two main issues.

1. Hand labelled data are often very expensive to acquire, as they require manual human curation. This limits the size of training datasets we can obtain.
2. Natural language, though very complicated, is also highly structured, and we should be able to learn some of this structure without the need for labels, and use it to improve our predictions.

Thus, as obtaining large amounts of unlabelled data is generally inexpensive, we can use this data to improve our classification algorithm without relying on the acquisition of more expensive labelled data.

These methods generally revolve around learning prior sentiment associations of words, known as unsupervised lexicon induction. This can be done in many ways(Riloff et al., 2003)(Kaji and Kitsuregawa, 2007), though the most common is through setting an initial set of “seed words” and then inducing sentiment

associations by how these seed words co-occur with others in the corpus. (Hu, 2004)(Turney, 2002)(Esuli and Sebastiani, 2005)(Gamon, 2005)(Pang and Lee, 2008)

This style of inference has been incorporated into both semi-supervised and unsupervised frameworks.

Chapter 2

Background

Recently, the creation of social media sites, such as Facebook and Twitter, has provided a new domain for the development and use of sentiment analysis techniques. These new sites bring new challenges and potential applications for sentiment analysis as they provide new dimensions of information, such as social network graphs and geographic information, and the style of text is in many cases very different than that of traditional domains such as reviews and articles. The scale and availability of such data is also far beyond that which is encountered in most traditional sentiment mining applications, bringing issues of computational speed and the ability to process documents in an online manner to light. The majority of my research has been performed on data from Twitter, due to the ease of accessing information through their public API. Because of this we will be mostly considering work relating to tweets (status message on Twitter), though the ideas and techniques apply more generally.

2.1 Labeled data

The issue of obtaining labeled data is worse in the social media context than in most others for sentiment analysis. Because the statuses are so short, we require many more labeled examples in order to have the same number of labels for each of the words in our vocabulary. Fortunately, due to the very large amount of data and easy search facilities in Twitter, we can propose to use a set of words as

noisy labels for different sentiments. The idea of using a set of initial keywords has been used in various forms in many papers.

In this framework, for each sentiment $s \in S$, we have a set of keywords K_s that we take as noisy labels for sentiment s . For example, K_{happy} could be *happy, glad, love, joy*. We then collect a set of tweets T_s by searching for tweets with these words contained in them. All these tweets are labeled “happy” and form part of our training set. The same process is repeated for the other sentiments and these sets of tweets are combined to form our full training set. We would like to be conservative in our selection of keywords for each sentiment, as choosing a keyword in this framework is equivalent to stating that any tweet containing the keywords will always be assigned to the sentiment for that keyword.

One candidate for generating such a list is the ANEW list of affective words (Bradley and Lang, 1999), which is a manually curated list of words, along with values that show how much they reflect a given sentiment, based on human tests. While this is a well constructed list, a major deficiency in the context of sentiment mining in social media is that it is a list of only English words. This means it is not appropriate for use in any system that considers non-English statuses.

Fortunately, the vocabulary of social media provides a small set of words that very strongly indicate sentiment, yet are language independent. These are emoticons, and were first used as noisy labels in (Go et al., 2009) and have subsequently been used in most systems for learning sentiment on Twitter.

2.2 Key traits of Twitter

There are several key traits of Twitter to consider when sentiment mining tweets. The main ones are the inherent multilingual nature of the service, the structure of tweets and the prevalence of spam.

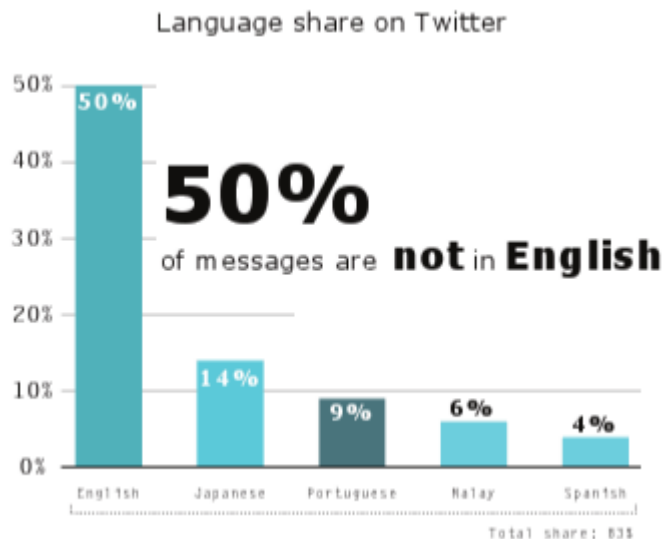


Figure 2.1: Prevalence of different languages on Twitter from (Semiocast, 2010).

2.2.1 Language diversity

While English is still the dominant language on Twitter, more than half of tweets are in a language other than English. Shown in Figure 2.1 is the distribution of language use in Twitter. While most research has focused purely on English tweets, there is obviously substantial value to methods that can work independent of language. However, casting the problem of sentiment analysis in a multi-lingual context prevents us from using information such as POS-tags and hard-coded negation rules as they are language dependent.

The work that has considered social sentiment mining in a multi-lingual context highlights the incredible advantage of using emoticons as noisy labels (Go et al., 2009), as they are practically¹ language independent.

¹There is some cross-cultural variation in use (Yuki et al., 2007), but these are small enough to manually specify.

Project Detox this week haha..crazzzy weekend!
OOOOOYYYYEEEEAA! Had a blast last night @ the O.B park wit tha hubby!.. Saw 2 many familiar faces....
YU HAVE A FREE COLLECT CALL FRM,, A FEDERAL FACILITY....*WINK* LMAO

Table 2.1: Examples of tweets.

2.2.2 Structure of tweets

Social network statuses have a distinctly different structure and feel to most other corpora (Table 2.1). As the context is very informal, there is frequent use of slang and specific web terminology. The statuses are usually very short, due both to the intended use of the service and the constraint that tweets cannot be more than 180 characters long. This leads to an average tweet length of approximately 11 words. The length of tweets is important in sentiment mining as it limits both how grammatically complex a status can be and more importantly, it restricts how much sentiment information can appear in a status. Because of this, it is very rare to encounter a single tweet that exhibits multiple sentiments. As a rough proxy for this, we can compare the rates of tweets with individual emoticons to those that contain both. Tweets with a single type of emoticon tend to be around 100-200 times more prevalent than those containing both happy and sad emoticons. Based on this, it seems a valid assumption that tweets have a single sentiment, and this greatly simplifies the class of models we have to consider to perform sentiment analysis.

2.2.3 Spam

One of the most difficult challenges for performing any kind of NLP on data from Twitter is overcoming spam. While Twitter is used by millions of normal individuals, our intended targets, it is also very popular among companies, marketers and other undesirable entities.

It is particularly problematic in our case, as marketers and spammers heavily use emoticons and so will almost certainly appear when looking for tweets

with emoticons. The other reason that this kind of spam is particularly bad for natural language applications is that the tweets are often templated or repeated verbatim under multiple accounts. While verbatim repetitions are easy to detect and ignore, templated tweets are not, and they have a dire effect on the kinds of models we are considering. A central assumption among nearly all sentiment models is that tweets are independent, which templated tweets break.

Because of the importance of detecting spam in this situation, there has been substantial research on this problem. Rates of 89% precision have been reported (Wang, 2010) on methods that use text and social graph features to detect spammers. However, due to technical constraints, the full social graph is not always available. In these cases, simpler rules have been developed that eliminate most spam, using only text features and follower counts.

The presence of a url is one of the strongest indicators of a spam tweet. This has a very high false positive rate, as there are obviously many legitimate tweets containing links. However by inspecting the tweet stream, we see that approximately 20% of tweets contain links. Thus by filtering out tweets with links we can remove most spam tweets while maintaining the same order of magnitude of the size of our dataset. A further measure is to ignore accounts with more than 1000 followers, or who follow more than 1000 accounts. This set of users was found to have a drastically different usage pattern to the average Twitter user (Kwak et al., 2010), indicating accounts such as news websites, which we are not interested in.

2.3 Predictive applications

It has been found that sentiment information from Twitter has predictive utility in a wide variety of areas. At the macro-scale, there has been work predicting a wide variety of trends based on sentiment information retrieved from Twitter. This has been in fields as varied as politics, marketing and finance. In (O'Connor et al., 2010a), it was found that correlations as high as 80% could be found between political sentiment data on Twitter and traditional polling methods. Recent

work has shown that Twitter sentiments are very strong predictors of movie box office performance ([Asur and Huberman, 2010](#)) and even the closing value of the Dow Jones Index ([Bollen et al., 2011](#)). This provides strong motivation to improve underlying sentiment analysis techniques, so that secondary systems such as these can be improved.

Chapter 3

A sentiment analysis method for Twitter

3.1 Motivation

In our literature review we saw that the most common indicator used to train sentiment classifiers on social network data are emoticons. All the techniques considered up to this point have been purely supervised learning ones. This means we can only learn information from training examples that contain one of our emoticon labels. However, we would assume that there are many happy and sads tweets that don't have an emoticon in them. We can try to extend our technique to use information in these other tweets to improve our model.

3.1.1 Mixture models

Mixture models provide a framework for unsupervised learning with categorical hidden variables. The idea behind a mixture model is that there are a number of discrete mixture components K, s_1, \dots, s_K . Associated with each of these components is a parameter vector θ_i , which defines the probability distribution for that component as $P(x|\theta_i)$, where P can be any probability distribution. While P could be a different distribution for different components, in most cases we consider it to be a single family of distribution. A graphical model for the process is shown in Figure 3.1.

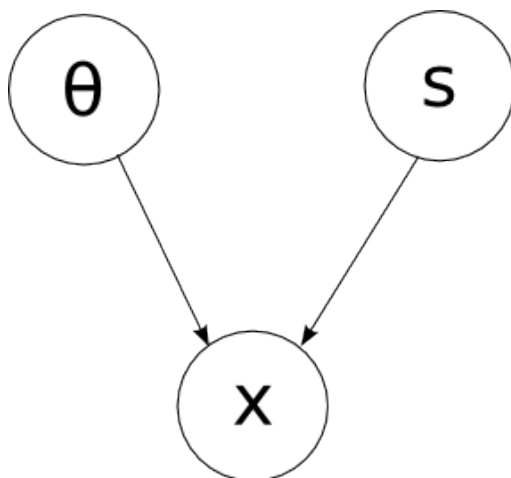


Figure 3.1: Graphical representation of a mixture model

In the case of our output domain being categorical and our input being a number of categoricals, a simple generative process is a mixture of multinomials model. In this situation a hidden label is drawn for an element i , and this element is then drawn from repeated draws from a categorical distribution.

The multinomial mixture-model is a good language model for tweets, as due to their short length they very rarely exhibit multiple sentiments. This is the justification for choosing this model over a more expressive model such as Latent Dirichlet Allocation (Blei et al., 2003).

3.1.2 Naive Bayes vs Mixture of Multinomials

By now it should be clear that there are strong similarities between the Naive Bayes model we outlined in the previous section, and the mixture of multinomial model. We can see that for both, the probability of a class label is:

$$p(y|x) = p(y) \prod p(x|y)$$

However, the difference is that in Naive Bayes, y is considered to be an observed variable, while in mixture of multinomials it is a hidden, or latent, variable.

Our data is in fact a combination of both types of data, with some tweets containing indicating words and some not. There are two different ways we can conceptualize this. The first, which we will use, is to consider the problem as completely unsupervised. Rather than having labels on our tweets (the emoticons), we have strong prior beliefs that these emoticons belong to a particular sentiment. However, this can equally be viewed as a semi-supervised learning problem where we maintain the assumption that the emoticons are labels and assume we have additional unlabeled data. Then we can apply techniques such as semi-supervised EM (Nigam et al., 2000), (Zhu, X. and Goldberg, 2009).

3.1.3 Bayesian mixture of multinomials

In a Bayesian setting, we need to have a prior belief distribution over our hidden variables, which are θ and s . We can choose this to be any distribution that reflects our beliefs. For simplicity, we take $P(\theta|\alpha)$ to be the conjugate prior of the categorical distribution, which is the Dirichlet distribution. This gives us a revised graphical model as shown in Figure 3.2.

The likelihood for the new model is:

$$p(y|x) \propto \prod p(y) \int p(x|\theta)p(\theta|\alpha)d\theta$$

In our case, we can now set the prior on θ to reflect our prior beliefs about what the word distribution should be for each sentiment. In this way, we are incorporating the correct information into our model in a principled way.

3.2 Sentiment prediction

Our model for sentiment prediction is as follows:

A set of tweets, \mathcal{T} , are generated from a multinomial mixture model, where the hidden mixture component s is the sentiment and the multinomial $\vec{\theta}_s$ is the word distribution for sentiment s . $\vec{\theta}_s$ is a vector of length $|V|$, that sums to 1, where each element $\theta_{s,i}$ is the probability that word w_i is drawn on a given draw from

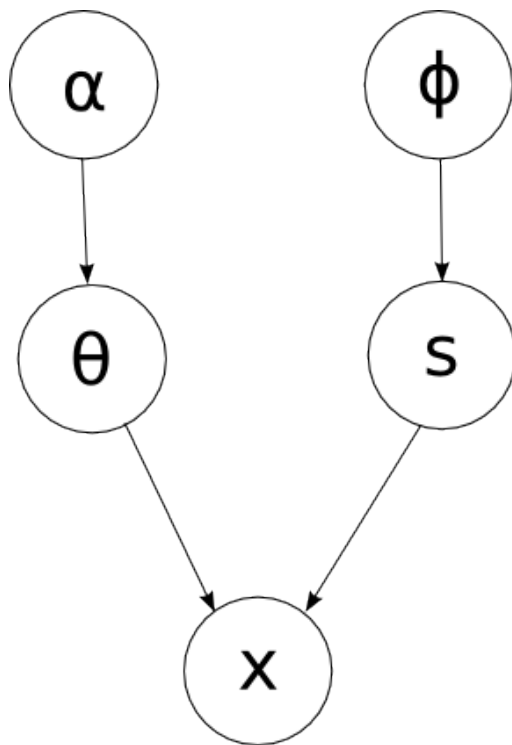


Figure 3.2: Graphical representation of a Bayesian mixture model

sentiment s . Our prior belief over each $\vec{\theta}_s$ is an asymmetric Dirichlet distribution $\text{Dir}(\vec{\alpha}_s)$. $\vec{\alpha}_s$ is based on a set of keywords for sentiment s , F_s , and will enforce that $\vec{\theta}_s$ is conceptually about sentiment s .

3.2.1 Likelihood

Our data consists of $|\mathcal{T}|$ tweets $t_i = (w_{i,1}, \dots, w_{i,n_i})$ where n_i is the number of words in tweet i . Each word is a member of the set V , which is the vocabulary of words we consider in our model. For each tweet t_i the probability of the tweet given a set of word distributions, one for each sentiment s , is:

$$p(t|\vec{\theta}) = \sum_{s \in \mathcal{S}} p(t|\vec{\theta}_s)p(s) \quad (3.1)$$

$$= \sum_{s \in \mathcal{S}} \prod_{w \in t} p(w|\vec{\theta}_s)p(s) \quad (3.2)$$

$$= \sum_{s \in \mathcal{S}} p(s) \prod_{w \in t} p(w|\vec{\theta}_s) \quad (3.3)$$

Also, the probability that a tweet belongs to a given sentiment is:

$$p(s|t) = p(t|s)p(s) \quad (3.4)$$

$$= \prod_{w \in t} p(w|s)p(s) \quad (3.5)$$

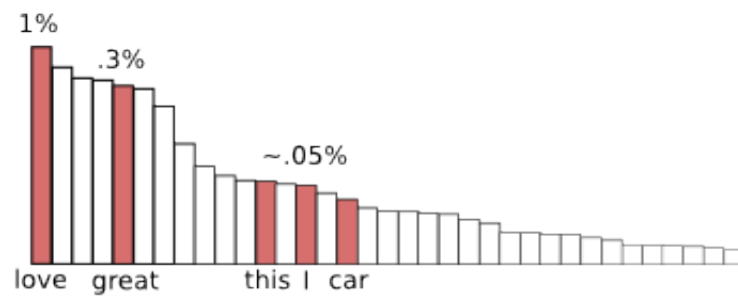
$$= p(s) \prod_{w \in t} \int_{\theta} p(w|\theta)p(\theta|\alpha_s) \quad (3.6)$$

An illustration of the likelihood function for two sentiment distributions, with elements highlighted for the phrase “I love this great car”, is shown in [3.3](#).

3.2.2 Prior

We need to construct our prior to encode our beliefs, as outlined at the beginning of the chapter. To do this we use an asymmetric Dirichlet prior $\text{Dir}(\vec{\alpha}_s)$. In a Dirichlet distribution, by setting an individual element to a higher value,

$$P(w|\theta_{\text{happy}})$$



$$P(w|\theta_{\text{sad}})$$

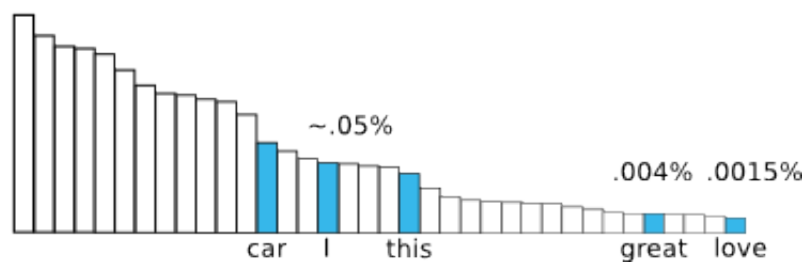


Figure 3.3: Example of likelihood function

we encode the belief that that element occurs more frequently in draws from the distribution θ . So, by setting a high value of $\vec{\alpha}_{\text{happy},i}$, where i is the index for the word ‘:)’’, we encode the belief that happy smilies are likely to occur in happy tweets. Similarly, by setting a low value of $\vec{\alpha}_{\text{sad},i}$, we encode the belief that happy smilies are unlikely to occur in sad tweets.

Formally, we set $\vec{\alpha}_s = k\vec{1} + \vec{\delta}_s$. All vectors are of length $|V|$, the size of the vocabulary. $\vec{1}$ is a vector with all values equal to 1, k is a scalar constant and $\vec{\delta}_s$ is defined as follows:

$$\delta_{s,w} = \begin{cases} \delta_+ & \text{if } w \in F_s \\ \delta_- & \text{if } w \in F_{s'} \text{ where } s' \neq s \\ 0 & \text{Otherwise} \end{cases} \quad (3.7)$$

δ_- and δ_+ are tunable parameters that are the weights added to the prior on the values of the indicative words. Setting $\delta_+ > 0$ adds mass to the indicative words for a sentiment, reflecting our belief that they are more likely, while $-\delta_- < 0$ removes mass from indicative words for other sentiments. The full form of the prior is thus:

$$p(\vec{\theta}_s) = \text{Dir}(k\vec{1} + \vec{\delta}_s) \quad (3.8)$$

Choice of δ_+

It is important to take some care in selecting the value of δ_+ . The value of δ_+ shouldn’t be set too high, as if the values of $\alpha_{s,i}$ for F_s are too high relative to the other values of $\vec{\alpha}_s$, our model would expect there to be multiple elements of F_s in a given tweet of sentiment s . It would then assign lower probabilities to tweets that have too few indicating words.

As a rule of thumb, it is best to set $\delta_+ < \frac{k|V|}{n|T_s|} \forall T_s$, where n is the average number of words in a tweet.

3.2.3 Learning

Now, rather than simply finding a maximum likelihood estimate of the parameters θ and s we wish to obtain a posterior distribution over these variables. Sadly, exact inference in MoM, or generally in mixture models, is very hard. For example, we may have 100 documents d_1, \dots, d_{100} . Even if we have only 3 hidden labels happy, sad, neutral, the number of ways these documents could be labelled is $3^{100} = 10^{50}$. To find the best possible assignment we could check each of these cases individually and evaluate which is the most likely. In general we can see the complexity of naive exact inference is $|S|^{|d|}$. There are no efficient methods for exact inference for these methods, so we must appeal to forms of approximate inference, that do not ensure an optimal answer, but usually find a good approximation in a reasonable amount of time.

A framework for inference that we can use is Variational Bayes (Ghahramani and Beal, 2000). Variational Bayes is a common method for efficiently finding an approximate distribution over model parameters in intractable Bayesian models. It finds this approximation by first assuming the distribution factorizes into a product of simpler distributions and iteratively optimizes each of the simpler distributions such that their product is closer to the true distribution we are trying to approximate.

3.2.3.1 Variational approximation

We would like to choose our model such that it maximizes the probability (or equivalently the log probability) of the data given the model $P(\theta|x)$. As we have discovered this is intractable for our model we can instead try and pick a restricted class of distributions Q and optimize within this set of distributions so we get an answer that is close to optimal.

One sensible metric is to minimize the Kullback-Leibler divergence between the variational distribution Q and the optimal P distribution. The KL divergence is defined as follows:

$$KL(Q||P(y, \theta|x)) = - \sum_y \int Q(y, \theta) \log \frac{P(y, \theta|x)}{Q(y, \theta)} d\theta$$

Also, we have the following identity:

$$\log P(x) = F(Q) + KL(Q||P(y, \theta|x))$$

where

$$F(Q) = \sum_y \int Q(y, \theta) \log \frac{P(y, \theta, x)}{Q(y, \theta)} d\theta$$

Because the log probability of x is independent of the parameters, we can minimize the KL divergence by maximizing the value of F .

We will need two probabilities for our later calculations:

$$\log P(y, x|\theta) = \log \prod_{s \in S} \prod_{w \in V} \theta_{s,w}^{n_{s,w}^y} \quad (3.9)$$

$$= \sum_{s \in S} \sum_{w \in V} n_{s,w}^y \log \theta_{s,w} \quad (3.10)$$

$$\log P(\theta|\alpha) = \log \prod_{s \in S} Dir(\theta_s|\alpha_s) \quad (3.11)$$

$$= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} - 1) \log \theta_{s,w} - c \quad (3.12)$$

where:

$$n_{s,w}^y = \sum_{i=1}^N \sum_{w' \in x_i} I[w' = w] I[y_i = s]$$

3.2.3.2 Mean-field

Our first approximation that we need to make is choosing the class of distributions Q that we will optimize over. A very common choice is the mean-field assumption, which is that the distribution factorizes into the product of separate terms, each of which only depend on a subset of the overall variables. In our case, we only

have two factors.

$$Q(y, \theta) = Q(y)Q(\theta)$$

Now,

$$F(Q) = \sum_y \int Q(y)Q(\theta) \log \frac{P(y, x, \theta)}{Q(y)Q(\theta)} d\theta$$

$$Q(y) = \frac{e^{E_\theta[\log p(y, x, \theta)]}}{\int e^{E_\theta[\log p(y, x, \theta)]}}$$

$$Q(\theta) = \frac{e^{E_y[\log p(y, x, \theta)]}}{\int e^{E_y[\log p(y, x, \theta)]}}$$

The solution to maximizing this equation involves setting the value of the log density of all the factors of Q to the expectation of $\log P(y, x, \theta)$ with respect to the other variables.

ie:

$$\log Q(y) = E[\log P(y, x, \theta)]_{Q(\theta)} - \log Z \quad (3.13)$$

$$= E[\log (P(y, x|\theta)P(\theta))]_{Q(\theta)} - \log Z \quad (3.14)$$

$$= E[\log P(y, x|\theta)]_{Q(\theta)} + (E[\log P(\theta)]_{Q(\theta)} - \log Z) \quad (3.15)$$

$$= E \left[\sum_{s \in S} \sum_{w \in V} n_{s,w}^y \log \theta_{s,w} \right]_{Q(\theta)} - \log Z \quad (3.16)$$

$$= \sum_{s \in S} \sum_{w \in V} n_{s,w}^y E[\log \theta_{s,w}]_{Q(\theta)} - \log Z \quad (3.17)$$

$$Q(y) = \frac{1}{Z} \prod_{s \in S} \prod_{w \in V} \theta'^{n_{s,w}^y} \quad (3.18)$$

$$= \prod_{t \in \mathcal{T}} \text{Multinomial}(\theta') \quad (3.19)$$

where

$$\theta'_{s,w} = \frac{e^{\psi(\alpha_{s,w} + n_{s,w})}}{e^{\psi(\sum_{w' \in V} \alpha_{s,w'} + n_{s,w'})}}$$

$$\log Q(\theta) = E[\log P(y, x, \theta)]_{Q(y)} - \log Z \quad (3.20)$$

$$= E[\log (P(y, x|\theta)P(\theta))]_{Q(y)} - \log Z \quad (3.21)$$

$$= E[\log P(y, x|\theta)]_{Q(y)} + E[\log P(\theta)]_{Q(y)} - \log Z' \quad (3.22)$$

$$= E[\log P(y, x|\theta)]_{Q(y)} + \log P(\theta) - \log Z' \quad (3.23)$$

$$= \log P(\theta) + E[\log P(y, x|\theta)]_{Q(y)} - \log Z' \quad (3.24)$$

$$\begin{aligned} &= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} - 1) \log \theta_{s,w} \\ &\quad + \sum_{s \in S} \sum_{w \in V} E[n_{s,w}^y]_{Q(y)} \log \theta_{s,w} - \log Z' \end{aligned} \quad (3.25)$$

$$\begin{aligned} &= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} - 1) \log \theta_{s,w} \\ &\quad + E[n_{s,w}^y]_{Q(y)} \log \theta_{s,w} \end{aligned} \quad (3.26)$$

$$= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} + E[n_{s,w}^y]_{Q(y)} - 1) \log \theta_{s,w} \quad (3.27)$$

$$\begin{aligned} &= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} + \\ &\quad E[\sum_{i=1}^N \sum_{w' \in x_i} I[w' = w] I[y_i = s]]_{Q(y)} - 1) \log \theta_{s,w} \end{aligned} \quad (3.28)$$

$$\begin{aligned} &= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} + \\ &\quad \sum_{i=1}^N \sum_{w' \in x_i} I[w' = w] E[I[y_i = s]]_{Q(y)} - 1) \log \theta_{s,w} \end{aligned} \quad (3.29)$$

$$\begin{aligned} &= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} + \\ &\quad \sum_{i=1}^N \sum_{w' \in x_i} I[w' = w] Q(y_i) - 1) \log \theta_{s,w} \end{aligned} \quad (3.30)$$

$$= \sum_{s \in S} \sum_{w \in V} (\alpha_{s,w} + \sum_{i=1}^N Q(y_i) \sum_{w' \in x_i} I[w' = w] - 1) \log \theta_{s,w} \quad (3.31)$$

$$= \text{Dirichlet}(\alpha_{s,w} + n'_{s,w}) \quad (3.32)$$

where

$$n'_{s,w} = \sum_{i=1}^N Q(y_i) \sum_{w' \in x_i} I[w' = w]$$

By iteratively calculating these values of Q , we will eventually converge to a locally optimal approximation to the full posterior.

3.2.4 Prediction

Once we have trained the model, the log probability of a tweet being generated given a particular sentiment can be computed as:

$$\log(p(t|\vec{\alpha}'_s)) = \sum_{w \in V} \sum_{i=0}^{n_{t,w}} \log(i + \alpha'_{s,w}) - \sum_{i=0}^{|t|} \log\left(i + \sum_{j=1}^{|V|} \alpha'_{s,j}\right) \quad (3.33)$$

Where $n_{t,w}$ is the number of occurrences of w in t . This is the log probability of an observation under a Dirichlet-Multinomial model. Using Bayes' rule and an empirical estimate of $p(s)$ this can be used to compute $p(s|t)$, which is our sentiment prediction.

$$p(s|t) = \frac{e^{\log(p(t|\vec{\alpha}'_s))}}{\sum_{s' \in S} e^{\log(p(t|\vec{\alpha}'_{s'})})} p(s) \quad (3.34)$$

Importantly, this can be calculated efficiently and is suitable for prediction in an online setting.

Chapter 4

Spatial priors

4.1 Motivation

We now return to the issue of sharing statistical strength between separate regions of data. A very natural set of regions for Twitter data is the geographic region they come from. More specifically, the geo-political hierarchy is a partitioning that should capture interesting diversity in the use of Twitter.

If we want to model separate sentiment distributions for different geographic regions, a first option would be to merely consider tweets from each region independently. However, by doing this we are disregarding any potential information we can learn about regions' word distributions from neighbouring regions. Especially in the cases where a region has very few tweets, it makes sense to use data from neighbouring regions to improve our estimates of the word distributions. We propose a simple, tractable extension to separately model word distributions while maintaining comparable classification accuracy when data is sparse.

A standard method for modeling this kind of spatially varying data in a Bayesian context is to use a hierarchical model such as the Hierarchical Dirichlet Process (Teh et al., 2006). In our case we can sacrifice some of the rigour and interpretability of this and similar models for reduced computation. In our case we don't need the extra power to discover common topics between documents, as

Country	% total tweet contributed
 USA	56.59
 UK	8.09
 Brazil	6.73
 Canada	4.36
 Australia	2.63
 Indonesia	2.34
 Germany	1.58
 Japan	1.47
 Netherlands	1.10
 India	0.97
 Mexico	0.90
 Singapore	0.88
 Philippines	0.85
 France	0.64
 Ireland	0.60
 Spain	0.57
 New Zealand	0.51
 Malaysia	0.47
 Italy	0.44
 Iran	0.41

Source: sysomos.com 

Figure 4.1: Global distribution of tweets (Sysomos, 2010).

we are already specifying our topics a priori through our priors. The main issue we wish to overcome is the data sparsity issue. This is because tweets are not uniformly distributed across the globe 4.1.

Due to this distribution of tweets, we can model word distributions very accurately in some tweet dense areas such as the US, but we have very little data in areas such as New Zealand to draw inference on.

4.2 Region hierarchy

We consider a hierarchy of regions R . We have a root node, r_0 , which contains all tweets. Below this, the child nodes of r_0 partition the tweets into disjoint subsets, with one belonging to each child node. These new sub-regions can in turn be subdivided into smaller subregions. In our case, the root node is “The World”. This has one child for each country of the world. Each country then has a child for each of its states/provinces. We define $N(r)$ as the neighbours of region r (those that share the same parent region), $P(r)$ as the parent of region r and T_r as the tweets contained in region r .

4.3 Prior

One way to incorporate data from neighbouring regions easily is to use it to form our prior for the region we are considering. It is not easy to directly incorporate the raw data from other regions into a prior, but if we had initial estimates of the word distributions from neighbouring regions then we could more easily construct our beliefs. To obtain this, we train our original model independently for each region to obtain a distribution over words, $p(\vec{\theta}_{s,r}|T_r) \forall r$. This means we have separate word distributions for each region, parametrized by $\vec{\alpha}_{s,r}$.

We would like our prior to obey the following restrictions:

1. Regions that are closer in the hierarchy have a stronger influence on the prior
2. There is a limit on the influence that any given region can have on the prior

We now want to define a prior $p(\vec{\theta}_{s,r}|T_{-r})$, which is our belief distribution over $\vec{\theta}_s$ in region r , given tweets observed in other regions.

In order to facilitate restriction 2, we introduce a function $f(x)$. This will be used to restrict the influence of neighbouring regions by changing the effective number of samples contributed towards the region from n to $f(n)$. We would like

this function to have a number of properties. Firstly, that there is an exponentially decreasing utility of additional points. This implies a form including the term $1 - e^{-x}$. Secondly, the function should asymptote at a value A , which we will take as a fixed parameter. This will mean that a neighbouring region can contribute at most effectively A samples. This implies a function of the form $A(1 - e^{-x})$. Lastly, we as we may be using different functions within the model with different values of A , we would like that $f(1) = 1$, so that an initial point is treated with equal weight, regardless of the overall maximum influence of a region.

$$f(1) = A(1 - e^{-1t}) = 1 \quad (4.1)$$

$$e^{-t} = 1 - \frac{1}{A} \quad (4.2)$$

$$-t = \log\left(\frac{A}{A} - \frac{1}{A}\right) \quad (4.3)$$

$$t = -\log\left(\frac{A-1}{A}\right) \quad (4.4)$$

$$t = \log(A) - \log(A-1) \quad (4.5)$$

This is graphed in Figure 4.2, to demonstrate how many effective samples would be used to influence neighbouring regions for a given number of samples x , given different choices of A .

$$f(x) = A \left(1 - e^{-x(\log(A) - \log(A-1))}\right) \quad (4.6)$$

The α parameter of a Dirchlet distribution can be interpreted as a vector of counts. In the case of a prior, we can interpret this as if we had previously observed a number of draws, which are described by the parameter vector.

Another parameterization to consider, is the vector α as $\alpha_{\text{prior}} + n\alpha'$, where n is $\sum_{i=0}^n \alpha_i - \sum_{i=0}^n \alpha_{\text{prior},i}$ and α' is $\frac{\alpha - \alpha_{\text{prior}}}{n}$. n is the number of samples observed, which can be interpreted as the confidence in the belief. α' will be equivalent to the maximum-likelihood estimate of θ .

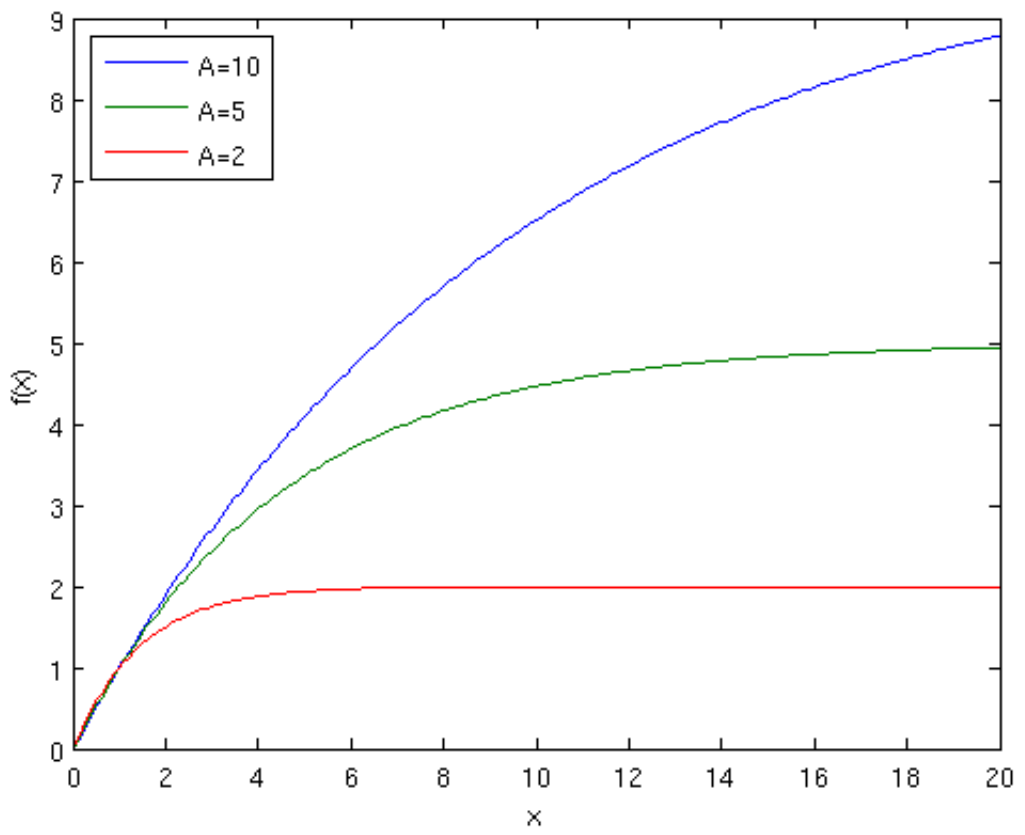


Figure 4.2: Thresholding function

In order to threshold the influence of an individual region, we make its contribution to the prior proportional to $f(n)\alpha'$. This limits the influence any given region can have on others prior to A samples. The prior counts are omitted as this would sum our prior beliefs for every region. We then take our prior to be a weighted average of this value for every region, weighted by its distance in the hierarchy. One way to achieve this is to define our Dirichlet parameter β as follows:

$$P(\vec{\theta}_{s,r}|T_{\neg r}) = Dir(\vec{\beta}_{s,r}) \quad (4.7)$$

$$\vec{\beta}_{s,r} = \omega_0 \frac{1}{|N(r)|} \sum_{r' \in N(r)} \frac{f(\sum_{i=1}^{|V|} \alpha'_{s,r',i})}{\sum_{i=1}^{|V|} \alpha'_{s,r',i}} \vec{\alpha}_{s,r'} + (1 - \omega_0) \vec{\beta}_{s,P(r)} \quad (4.8)$$

ω_0 is an additional free parameter that trades off the amount that directly neighbouring regions and more distance neighbours influence the distribution. This can be understood as seeing $\vec{\beta}_{s,r}$ as a weighted average of the influence of its direct neighbours (the first term) and the influence of all other regions (the second term, which is the β for the parent region). Since this is recursively defined, this means that the influence of regions are weighted by $(1 - \omega_0)^k \omega_0$ where k is the distance within the hierarchy.

Now we can train a new model that is the same as the model that considered the regions independently, only now we replace the priors with those $p(\vec{\theta}_{s,r}|T_{\neg r})$ that we just calculated.

Thus our final procedure for incorporating spatial information is as follows:

1. Train independent models for each region r , as described in Chapter 3.
2. For each region r calculate $\vec{\beta}_{s,r}$, as described above.
3. Train models for each region r in the same way as Step 1, but replacing the priors for each region with $\vec{\beta}_r$

Chapter 5

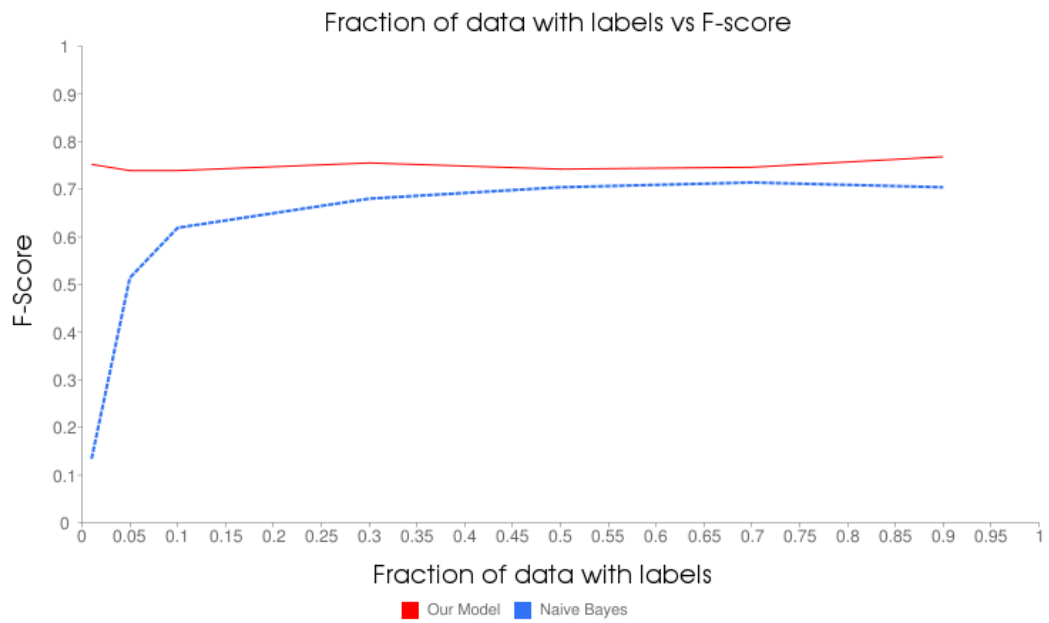
Experiments

To evaluate our models, we perform two sets of experiments; one on the proposed sentiment model outlined in Chapter 3 and the second on the proposed geographical extension outlined in Chapter 4.

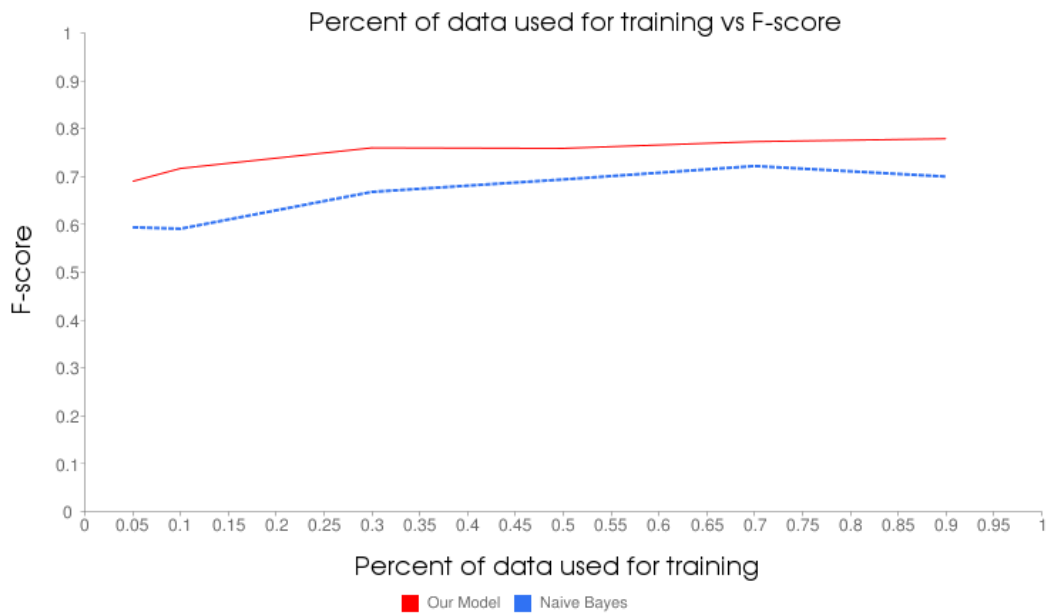
To evaluate our sentiment model, we compare against a Naive Bayes classifier which, as outlined previously, is the most consistently well-performing sentiment classifier. In our comparison against Naive Bayes we test using a sample of 50,000 tweets with a happy or sad emoticon. 60% of these contain a happy emoticon, while 40% contain a sad emoticon. No tweet contains both a happy and sad emoticon.

To evaluate our geographic model, we assess the improvement over the non-geographic version of our algorithm. In our geo model comparison, we test on a different sample of 20,000 tweets with happy or sad emoticon that also include geo information.

For illustrative purposes, we also include an example list of high probability words for different regions and sentiments (Table 5.1) and maps (Figures A.5, A.2) showing relative rates of happy and sad tweets for different regions of the world.



(a) Effect of removing labels from training data



(b) Effect of altering percent train/test split

Figure 5.1: Comparison of our model with Naive Bayes

Region	Sentiment	Top words
World	Happy	amore amour liebe happy :-) kasih amores bahagia love feliz :) makasih follback seguindo oooi good follow lovely xxx selamat hey s/o terima cheers thx
	Sad	sedih :-(sad triste :(dor saudade droga fome cade doendo aaa saudades garganta mimimi aff morreu odeio gripe perdi merda aaah :'(
UK	Happy	amour feliz kasih :-) happy love :) thanks birthday thank follow good welcome luck great nice morning amazing hello lovely xxx best hey awesome cheers
	Sad	:-(sad triste :(nooo booo </3 poor miss *cries* gutted afford :'(poorly hate dean ugh fml urghhh stressed headache *sigh* canada prayforkatie richards rip whyyy horrible revision *hugs*

Table 5.1: List of highest probability words for each sentiment at different regions

5.1 Sentiment model comparison

Firstly, we test the accuracy of the proposed tweet model without geographic information and for comparison we use a Naive Bayes classifier trained on the same features. For the first test we take 50% of the data for training and from a fraction of these tweets, remove the emoticons. These represent emotive tweets that we would encounter on Twitter that do not have an emoticon. The models are trained on this data and then the classification accuracy is assessed on the remaining 50% of the data, with all emoticons removed. The accuracy measure used is the F-score of retrieving happy tweets. The F-score is defined as:

$$F = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

which can be interpreted as a harmonic mean of the precision and recall.

In Figure 5.1 we see that when very few labelled training examples are supplied, our model far outperforms Naive Bayes. In the case where very few labels are supplied, the F-score for Naive Bayes is barely above the baseline of .375. This baseline comes about as we have 60% happy tweets in our dataset and we

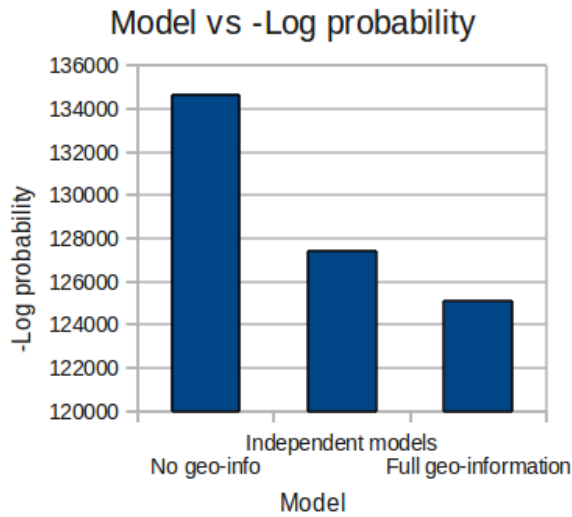
can achieve a recall of 1 and precision of .6 by simply classifying every tweet as happy. Our model does not increase performance substantially as labels are added, but Naive Bayes increases to just below ours.

5.2 Effect of incorporating geo-information

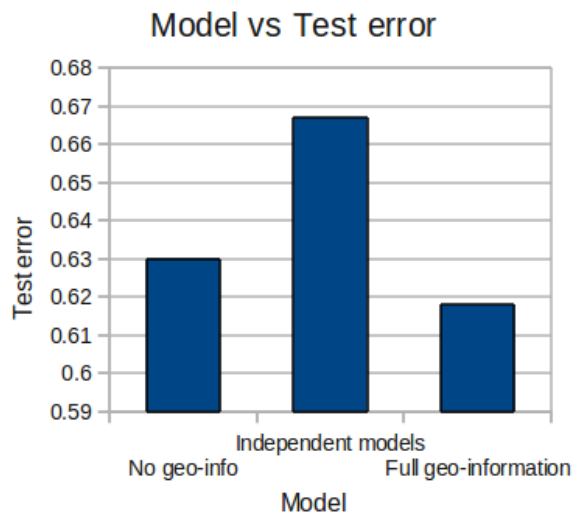
For the second test we remove emoticons from 90% of the training data, but alter the fraction of data used for training.

Secondly, we look at the effect of incorporating geo-information into our model. The three models we test are our model without geo-information, independent instances of our model for each region and incorporating neighbouring region information as outlined in Section 4.2. We compare them based on classification error and the negative log probability of the test dataset under the model. Both of these are averaged over 10 different test splits. The negative log probability gives us an idea of how well the data is modelled, as a lower value of this means that the model was less “surprised” by the data.

We can see in Figure 5.2 that by training independent models for different regions, we lose classification accuracy, even though we are modelling the data better, as shown by the lower negative log probability. However, by incorporating neighbouring region information, we recover our classification accuracy and achieve an even further reduction in negative log probability.



(a) Negative log probability of test data under different models (smaller is better)



(b) Test error under different models

Figure 5.2: Comparison of our model without geo information, with independent partitioning and using neighbouring geo-information

Chapter 6

Future plan for Ph.D.

Main goals

1. Develop theory around nested sparse approximations (NITC)
2. Implement high-performance distributed GPs for large scale datasets
3. Create GP-light (compiled C version of GPML)

Tasks

October 2011 - January 2012

Task	Estimated time	KPI
Write base GP-light (with DD)	4 weeks	Working executable
Specify NITC approximation	4 weeks	Written up on paper
Investigate GP-light use case and default parameters (with DD)	1 week	Defaults flow chart implemented in exe
Set-up test framework for SVM vs GP	1 week	Working test framework
Collect datasets	2 day	50 real world datasets in common format
Construct datasets	1 week	10 constructed datasets that reflect differences in SVM and GP
Decide on comparison methodology	1 week	Methodology specification document with justification
Full complexity analysis of NITC calculations	1 week	Written up on paper
Literature review of sparse GP approximations	3 weeks	Written up literature review
Create web page for GP-light	2 days	Host live site
Write popular academic articles arguing for the use of GPs	1 week	2 published articles in high profile popular academic areas

February 2012 - May 2012

Write robust classification likelihood	2 weeks	Implement in GPML
Publish GP vs SVM paper	3 weeks	Publish paper
Analyse covariance function distribution for common covariance functions/datasets	1 month	Written up analysis
Write up basic NITC implementation	2 weeks	Matlab code
Write NITC paper	1 month	Submit paper
Investigate clustering techniques for forming heirarchy.	1 month	Literature review of clustering techniques
Investigate parallel computing frameworks	1 week	Report comparing frameworks

June 2012 - September 2012

Write high-performance parallel NITC implementation	3 months	Working implementation scaling to 10^7 points
Write paper for SIGKDD about scaling GPs	3 weeks	Written paper

October 2012 - January 2013

Investigate extending framework to non-gaussian likelihoods

February 2013 -May 2013

Investigate extensions (transductive support point selection/online GPs/active learning)

June 2013 -September 2013

Finish write-up

Appendix A

System implementation

Data acquisition is a non-trivial step, involving filtering of users, filtering of tweets, parsing of tweets and determining the region of tweets.

A.1 Obtaining data

Twitter provides two APIs to access information about tweets; the Search API and the Streaming API. When selecting an API to use, there are several important characteristics to consider.

APIs

The Search API is intended to provide the ability to perform specific, low throughput queries. One can refine by user, content, geographic location (defined by a GPS co-ordinate and radius) and date. Importantly, only tweets from the preceding 5 days can be searched and queries are limited to approximately 10 per minute at the time of writing.

The Streaming API is designed to allow access to a live stream of tweets, as they occur. One can refine by user, content and area (defined by a rectangular bounding box). However, the user/content filter and the geographic filter are performed with a logical “OR”. This means that a search for tweets “Charlie” in San Francisco will return all tweets from San Francisco and all tweets containing “Charlie”. We are interested in predicting on the streaming API.

text	The body text of the tweet
id	The unique id of the tweet
author	The author of the tweet
retweeted	A boolean variable indicating if this is a retweet
coordinates	Co-ordinates the tweet originated from (if available)
followers	The number of followers the user has
following	The number of users that the user follows
place	Geo information, (place name, bounding box)
created at	Timestamp the tweet was created at

Table A.1: Fields returned in Twitter API

To collect a dataset of substantial size quickly we use the Streaming API, though the Search API is also used to compile testing datasets.

Data format

When querying either API, we are returned a list of status updates. Each status contains fields describing the tweet. The fields we are concerned with are shown in Table A.1.

Some other information is returned, mainly relating to whether a tweet is in response to another tweet or user. Through another API call, more information can be requested about a particular user. However, the search API’s rate limit is too low to query every user we see in the stream. Therefore for prediction on the stream, we cannot rely on having access to this extra data.

A.2 Filtering Tweets

Spam accounts

While there are millions of legitimate users on Twitter, there are also a large number of spam accounts, corporate accounts and bots (most notably weather bots), that have a very negative effect on standard language models. This is due both to the volume of tweets they generate, which is generally much higher than legitimate users, and the fact that many of the tweets are automatically generated or templated. This leads to many words artificially appearing together many

times, which breaks the assumptions of language models based on co-occurrences such as ours.

Identification of spam accounts is a separate research question in itself (Chu et al., 2010)(Stringhini et al., 2010)(Wang, 2010). However, as noted before, in the online setting we are severely limited in the number of features we have for a given user. We don't have access to their full feed or social graph. The only informative features are the number of followers that a user has, as well as the number that they are following. Research has shown that users with more than 1000 followers or who follow more than 1000 users have a drastically different usage pattern to normal users (Kwak et al., 2010). This may indicate that the account is spam, a celebrity, a corporation or other accounts we are not interested in. Therefore as a simple rule we filter out all these users.

Duplicate tweets

An important aspect of Twitter that breaks the assumption of most language models are retweets. Retweets are when a user rebroadcasts a tweet that was tweeted by another user. This is usually of the form: "RT @[Original User] [Original Text]". While the fact that a user has retweeted another tweet is likely to contain meaningful sentiment information, the fact that the words appear verbatim strongly breaks the iid assumption of common language models. While retweeting is an important part of Twitter culture, we choose simply to remove the retweets at sampling time.

A.3 Tweet Processing

Once we have retrieved our tweets from the API, we convert them from the string representation to a feature representation. For our representation we choose a bag-of-words model.

Tokenization

Tweets are not written like other text documents. The 140 character limit and informal setting result in tweets that heavily use slang, emoticons, spelling and

punctuation that would not be found in traditional text documents. Because of this, we need to perform custom pre-processing and tokenization of the tweets. We use a tokenizer specifically designed for Twitter (O'Connor et al., 2010b). This much better captures the use of punctuation, emoticons and words on Twitter. Additional steps were to conflate more than three consecutive occurrences of the same letter down to two letters. There are very few meaningful terms that contain more than three of the same character in a row, but it is very common on Twitter to repeat letters for emphasis.

A.4 Determining location in geo-hierarchy

As already mentioned in the previous section, the geographic information may come as either the GPS co-ordinates that the tweets was tweeted from, or the name of a location with a bounding polygon for that region. When creating a system to model regional variation, we would like to use the geopolitical region of the tweet. Fortunately, there is an API provided by GeoNames.org that converts GPS co-ordinates to a code defined in ISO 3166-2. ISO 3166-2 are two region codes separated by a dash that are the country code and a sub region (state/province) code.

This service is also rate-limited such that we cannot query at the rate of the streaming API. However, if we receive a tweet's location information as a name and bounding region pair, we can calculate a representative GPS co-ordinate within the region, query the GeoNames API and cache the result against the location name. As a representative point we use the centroid. While this is not assured to be inside the bounding polygon, as the polygons can be non-convex, in practice this works for almost all points.

A.5 Plotting

The plotting for the user interface was performed using Google's Chart API (Google). Examples of this are shown in A.5 and A.2. Regions with no tweets are left blank, regions with data are ranked based on the predicted ratio of happy to sad tweets.

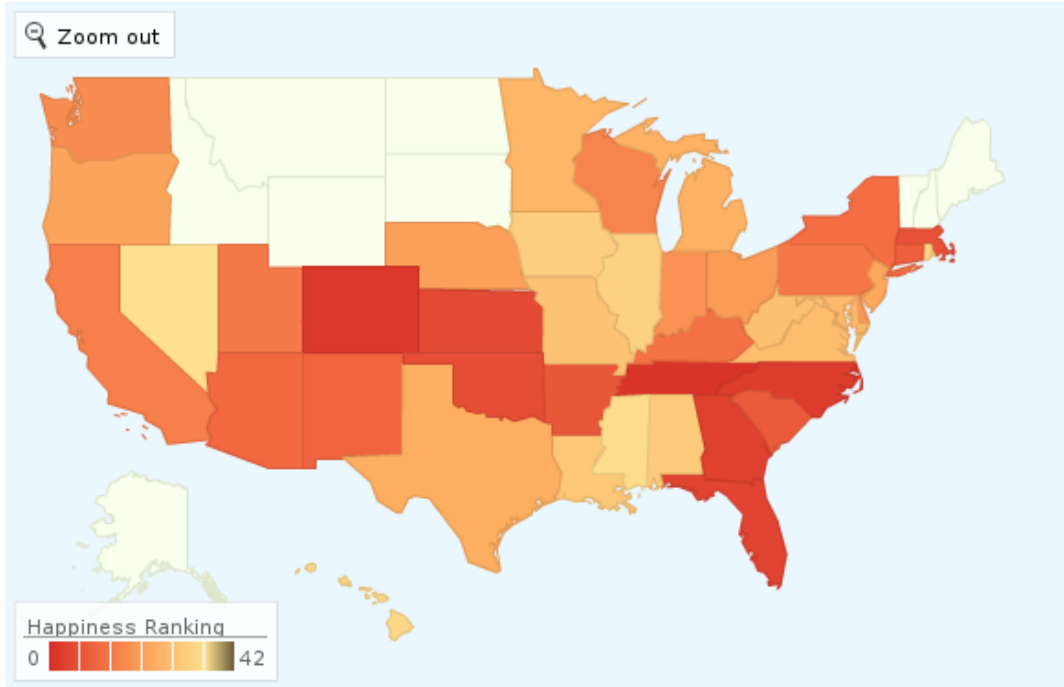


Figure A.1: Relative rates of happy to sad tweets by country.

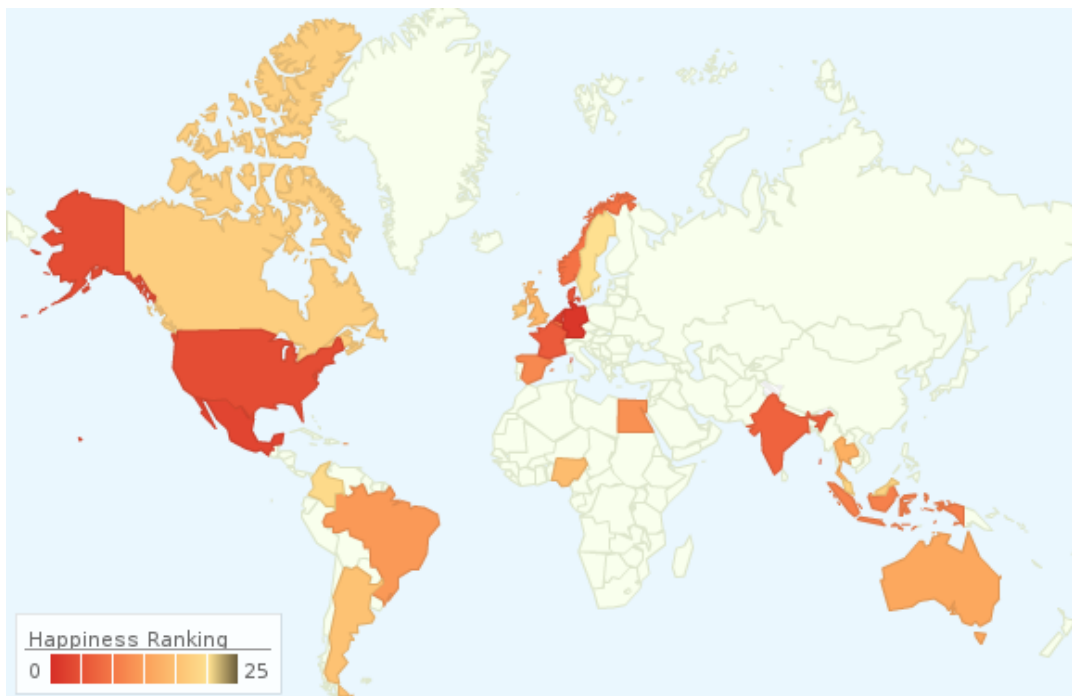


Figure A.2: Relative rates of happy to sad tweets by US State.

Bibliography

- Sitaram Asur and B.A. Huberman. Predicting the future with social media. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 492–499. IEEE, 2010. URL <http://www.computer.org/portal/web/csdl/doi/10.1109/WI-IAT.2010.63>. 15
- Luciano Barbosa and J. Feng. Robust sentiment detection on Twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, number August, pages 36–44. Association for Computational Linguistics, 2010. URL <http://portal.acm.org/citation.cfm?id=1944571>. 5, 7
- M.J. Beal and Zoubin Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics 7*., 2003. URL http://books.google.com/books?hl=en&lr=&id=fbErU8g_MLEC&oi=fnd&pg=PA453&dq=The+Variational+Bayesian+EM+Algorithm+for+Incomplete+Data:+with+Application+to+Scoring+Graphical+Model+Structures&ots=zZhpZXQwpN&sig=q1BH6AupHH1kGwt0BtCPahw8Ch4.
- Steven Bird, Ewan Klein, and Edward Loper. Natural Language Toolkit: Categorizing and Tagging Words. URL <http://nltk.sourceforge.net/doc/en/ch03.html>. 5
- David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18(1):147, June 2006. ISSN 1932-6157. doi: 10.1214/07-AOAS114. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.61.2352&rep=rep1&type=pdf>.

- David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, May 2003. ISSN 1532-4435. doi: 10.1162/jmlr.2003.3.4-5.993. URL http://www.crossref.org/jmlr_DOI.html. 17
- David Blei, T.L. Griffiths, Michael Jordan, and Josh Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. *Advances in neural information processing systems*, 16:106, 2004. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.113.7572&rep=rep1&type=pdf>.
- Johan Bollen, Huina Mao, and Xiao-jun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, pages 1–8, 2011. URL <http://linkinghub.elsevier.com/retrieve/pii/S187775031100007X>. 15
- M M Bradley and P J Lang. Affective norms for English words (ANEW): Stimuli, instruction manual and affective ratings. Technical report, The Center for Research in Psychophysiology, University of Florida, 1999. 11
- Claire Cardie, Janyce Wiebe, Theresa Wilson, and Diane Litman. Combining Low-Level and Summary Representations of Opinions for Multi-Perspective Question Answering. Technical report, Association for the Advancement of Artificial Intelligence, 2003.
- Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing - HLT '05*, (October):355–362, 2005. doi: 10.3115/1220575.1220620. URL <http://portal.acm.org/citation.cfm?doid=1220575.1220620>.
- Zi Chu, Steven Gianvecchio, Haining Wang, and S. Jajodia. Who is tweeting on Twitter: human, bot, or cyborg? In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 21–30. ACM, 2010. URL <http://portal.acm.org/citation.cfm?id=1920265>. 44

- Kushal Dave, Steve Lawrence, and D.M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM, 2003. URL <http://portal.acm.org/citation.cfm?id=775226>.
- L Dini and G Mazzini. Opinion classification through information extraction. In *Intl. Conf. on Data Mining Methods and Databases for Engineering, Finance and Other Fields*, pages 299–310. Citeseer, 2002. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.109.1736&rep=rep1&type=pdf>.
- J. Eisenstein, Brendan O’Connor, N.A. Smith, and E.P. Xing. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287. Association for Computational Linguistics, 2010. URL <http://portal.acm.org/citation.cfm?id=1870782>.
- Andrea Esuli. Determining term subjectivity and term orientation for opinion mining. *Proceedings the 11th Meeting of the Conference of the European Chapter of the Association for Computational Linguistics*, 2(1):193–200, 2006. URL http://acl.ldc.upenn.edu/eacl2006/main/papers/13_1_esulisebastiani_192.pdf.
- Andrea Esuli and Fabrizio Sebastiani. Determining the semantic orientation of terms through gloss classification. *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM ’05*, page 617, 2005. doi: 10.1145/1099554.1099713. URL <http://portal.acm.org/citation.cfm?doid=1099554.1099713>. 9
- Michael Gamon. Automatic identification of sentiment vocabulary: exploiting low association with known sentiment terms. *Proceedings of the ACL Workshop on Feature*, 2005. URL <http://portal.acm.org/citation.cfm?id=1610241>. 9
- Zoubin Ghahramani and M.J. Beal. Variational inference for Bayesian mixtures of factor analysers. *Advances in neural information processing systems*, 12:

- 449–455, 2000. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.4497&rep=rep1&type=pdf>. 23
- Alec Go, Richa Bhayani, and Lei Huang. Twitter Sentiment Classification using Distant Supervision. Technical report, Stanford University, 2009. 11, 12
- Google. Google Chart Tools: GeoChart. URL <http://code.google.com/apis/chart/interactive/docs/gallery/geochart.html>. 45
- Minqing Hu. Mining opinion features in customer reviews. *Proceedings of the National Conference on Artificial Intelligence*, 2004. URL <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Mining+Opinion+Features+in+Customer+Reviews#0>. 9
- Nobuhiro Kaji and Masaru Kitsuregawa. Building lexicon for sentiment analysis from massive collection of HTML documents. In *Proceedings of the joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, number June, pages 1075–1083, 2007. URL <http://acl.ldc.upenn.edu/D/D07/D07-1115.pdf>. 8
- Slava Katz. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-35(3), 1987. 4
- Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a Social Network or a News Media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591—600, 2010. 14, 44
- John Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 282–289. Citeseer, 2001. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.23.9849&rep=rep1&type=pdf>.
- Thomas Lake. Twitter Sentiment Analysis. Technical report, Western Michigan University, April 2011.

- W. Li and A. McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584. ACM, 2006. URL <http://portal.acm.org/citation.cfm?id=1143917>.
- Christopher Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999. 4
- S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. Mining product reputations on the web. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 341–349. ACM, 2002. URL <http://portal.acm.org/citation.cfm?id=775098>.
- Tetsuya Nasukawa and Jeonghee Yi. Sentiment Analysis : Capturing Favorability Using Natural Language Processing Definition of Sentiment Expressions. *Proceedings of the 2003 International Conference on Knowledge Capture*, pages 70–77, 2003.
- Kamal Nigam, A.K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2):103–134, 2000. URL <http://www.springerlink.com/index/p4324q3673265225.pdf>. 18
- Brendan O’Connor, Ramnath Balasubramanyan, Bryan Routledge, and Noah A. Smith. From tweets to polls: Linking text sentiment to public opinion time series. In *Fourth International AAAI Conference on Weblogs and Social Media*, number May, 2010a. URL <http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/viewPDFInterstitial/1536/1842>. 14
- Brendan O’Connor, Michel Krieger, and David Ahn. TweetMotif: Exploratory search and topic summarization for twitter. *Proceedings of ICWSM*, (May):2–3, 2010b. URL <http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/viewPDFInterstitial/1540/1907>. 45
- Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. *Proceedings of LREC 2010*, pages 1320–1326,

2010. URL <http://deephoughtinc.com/wp-content/uploads/2011/01/Twitter-as-a-Corpus-for-Sentiment-Analysis-and-Opinion-Mining.pdf>. 7
- Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004. URL <http://portal.acm.org/citation.cfm?id=1218990>.
- Bo Pang and Lillian Lee. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1):1–135, 2008. ISSN 1554-0669. 1, 2, 4, 9
- Bo Pang, Lillian Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002. URL <http://portal.acm.org/citation.cfm?id=1118704>. 5, 7
- C.E. Rasmussen and Zoubin Ghahramani. Infinite mixtures of Gaussian process experts. *Advances in neural information processing systems 14: proceedings of the 2002 conference*, 2:881, 2002. URL http://books.google.com/books?hl=en&lr=&id=GbC8cqXGR7YC&oi=fnd&pg=PA881&dq=Infinite+Mixtures+of+Gaussian+Process+Experts&ots=ZvJ0H31wu5&sig=Tg9ARvg_2Jin2RaT0fQVoapscNQ.
- Ellen Riloff, Janyce Wiebe, and Theresa Wilson. Learning subjective nouns using extraction pattern bootstrapping. *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, 4:25–32, 2003. doi: 10.3115/1119176.1119180. URL <http://portal.acm.org/citation.cfm?doid=1119176.1119180>. 8
- SemioCast. Half of messages on Twitter are not in English, 2010. URL http://semioCast.com/downloads/SemioCast_Half_of_messages_on_Twitter_are_not_in_English_20100224.pdf. 12

- Gianluca Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 1–9. ACM, 2010. URL <http://portal.acm.org/citation.cfm?id=1920263>. 44
- Sysomos. Exploring the Use of Twitter Around the World, 2010. URL <http://www.sysomos.com/insidetwitter/geography/>. 29
- Jie Tang. An Introduction for Conditional Random Fields. Technical report, 2005.
- Yee Why Teh, Michael Jordan, M.J. Beal, and David Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101 (476):1566–1581, 2006. URL <http://pubs.amstat.org/doi/abs/10.1198/016214506000000302>. 28
- Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, 1:173–180, 2003. doi: 10.3115/1073445.1073478. URL <http://portal.acm.org/citation.cfm?doid=1073445.1073478>. 4
- P. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, number July, pages 417–424, 2002. URL <http://nparc.cisti-icist.nrc-cnrc.gc.ca/npsi/ctrl?action=shwart&index=an&req=8914166&lang=en>. 9
- A.H. Wang. Don't follow me: Spam detection in Twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–10. IEEE, 2010. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5741690. 14, 44
- Janyce Wiebe, Eric Breck, Chris Buckley, Claire Cardie, Paul Davis, Bruce Fraser, Diane Litman, David Pierce, Ellen Riloff, Theresa Wilson, and Others.

BIBLIOGRAPHY

- Recognizing and organizing opinions expressed in the world press. In *Working Notes-New Directions in Question Answering (AAAI Spring Symposium Series)*, 2003. URL <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Recognizing+and+Organizing+Opinions+Expressed+in+the+World+Press#0>.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis. *Computational Linguistics*, 35(3):399–433, September 2009. ISSN 0891-2017. doi: 10.1162/coli.08-012-R1-06-90. URL <http://www.mitpressjournals.org/doi/abs/10.1162/coli.08-012-R1-06-90>. 5
- M Yuki, W Maddux, and T Masuda. Are the windows to the soul the same in the East and West? Cultural differences in using the eyes and mouth as cues to recognize emotions in Japan and the United States. *Journal of Experimental Social Psychology*, 43(2):303–311, March 2007. ISSN 00221031. doi: 10.1016/j.jesp.2006.02.004. URL <http://linkinghub.elsevier.com/retrieve/pii/S0022103106000321>. 12
- A.B. Zhu, X. and Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3:1–130, 2009. 18