Entity Based Sentiment Analysis on Twitter

Siddharth Batra and Deepak Rao
Department of Computer Science
Stanford University
(sidbatra,drao)@cs.stanford.edu

1   Introduction

Its fair to say that when it comes to global entities that matter to us, we are quite obsessed with finding out what other people think about them. In the past, querying people's opinions with respect to a global entity such as Michael Jackson, Microsoft or Cuba was done through user polls funded by organizations and only a small percentage of these questionable polls trickled to the average person. With the onset of real-time information networking service Twitter [5] there is an unparalleled public collection of opinions about every global entity that matters. Opinion creation has never been so democratized. Even though Twitter provides a great outlet for opinion creation, the information cycle is incomplete without clever tools for analyzing those opinions to facilitate their consumption.

The area of sentiment analysis [13] aims to understand these opinions and distill them into discrete categories such has happy,sad,angry or more simply positive, negative,neutral. Sentiment analysis has been popular since the start of this decade primarily fueled by the desire to understand and classify user reviews [9, 10, 11, 12]. Even though smaller datasets were used in the beginning the trend has been increasingly moving towards using large corpora of data.

To mine Twitter for entity opinions, we have used a dataset of Tweets (Twitter messages) spanning two months starting from June 2009. The dataset has roughly 60 million tweets. The entire dataset has been prepared by the Stanford InfoLab [16] and contains every Tweet sent from June - Dec of 2009. Below is an example of a single tweet in the dataset which comes with the time at which it was sent and the user who sent it.

```
T   2009-07-31 23:59:58
U   http://twitter.com/petegrillo
W   I find Facebooks UI to be so confusing. My FB stream in
    #tweetdeck is different than the news stream in FB.
    How can that be?
```

The aim of our work is to use the Twitter corpus to ascertain the opinion about entities that matter and enable consumption of these opinions in a user friendly way. We focus on classifying the opinions as either positive, negative or neutral. Since there aren't large enough datasets of labeled tweets, limiting the sentiment categories to the above three enables us to leverage other similar but larger datasets for training custom sentiment language models.

We begin by extracting entities from the Twitter dataset using the Stanford NER [8]. URLs and username tags (@person) are also treated as entities to augment the entities found by the NER.

To learn a sentiment language model we use a corpus of 200,000 product reviews that have been labeled as positive or negative. Using this corpus the sentiment language model computes the probability that a given unigram or bigram is being used in a positive context and the probability that its being used in a negative context.

Using this sentiment language model we analyze all tweets associated with an entity and classify whether the overall opinion of that entity is positive or negative and by how much.

We also provide intuitive ways for consuming these sentiments. Individual tweets are classified and ranked enabling easy consumption of some of the most positive and negative opinions. Tweets are analyzed on a timeline to show how the opinion of the entity has changed over a period time, perhaps to reflect the impact of certain events. Lastly, we provide an intuitive opinion cloud that visualizes the distribution over the frequency of the words associated with an entity and their label - positive or negative.

The following sections will present theoretical and implementation details along with supporting results for each of the modules summarized above. Towards the end the Results section will present a selection of fascinating results that showcase the impact of classifying and visualizing people's sentiment towards global entities.

2 Named Entity Recognition

Since opinions are usually targeted at an entity we begin by extracting entities from the Twitter corpus. An NER labels sequences of words in which there are names of entities such as people, organizations , locations etc. We utilize the Stanford NER [8] to extract such named entities.

Below is an example of a tweet and the entities extracted by the Stanford NER.

```
@billgates <ORGANIZATION>Microsoft</ORGANIZATION> expects tough market
for Windows 7: <ORGANIZATION>Microsoft</ORGANIZATION>
executive <PERSON>Bill Koefoed</PERSON> answered some
questions about the.. http://tinyurl.com/l4a4z9
```

The Stanford NER can't automatically classify names tags (@person) and URLs as entities so we augment it by including these types of entities. It seems intuitive to collect person tags since this is how Twitter users express their opinion or communicate with the entity being tagged. Similarly, opinions about URLs that are shared on Twitter, since disseminating URLS is of the chief use cases for Twitter.

```
<NAME>@billgates</NAME> <ORGANIZATION>Microsoft</ORGANIZATION>
expects tough market for Windows 7: <ORGANIZATION>Microsoft
</ORGANIZATION> executive <PERSON>Bill Koefoed</PERSON>
answered some questions about the..
<URL>http://tinyurl.com/l4a4z9</URL>
```

The NER implementation comes with four pre-trained classifiers. The first type has been trained on data from CoNLL, MUC6, MUC7, and ACE and since it is trained on data from the US and UK newswire it is robust across both domains. It classifies entities into PERSON, ORGANIZATION and LOCATION. The second type has been trained on data from CoNLL dataset and classifies entities into PERSON, ORGANIZAION, LOCATIAON and MISC. Each of these classifiers comes with a second version that uses a distributional similarity lexicon [8] to improve performance. We used the former three class version with the distributional similarity lexicon.

It is interesting to note that running even the most trivial of operations on a dataset of 250 million lines (3-4 lines of meta data per tweet) has its own computational challenges. For running the NER on the Twitter corpus we used a home-grown Map-Reduce approach to utilize the four cores of the quad-core machine available to us.

3 Sentiment Language Model

With the sentiment language model our aim is to understand the connotation with which a word is used within a tweet. After computing the sentiment language model we obtain the probability that a given token is being used in a positive context and the probability that a given token is being used in a negative context.

Given the short nature of tweets, we found that a lot of users make heavy use of emoticons to express themselves. Our model has special tokens that take into account a broad range of emoticons.

The following sections analyze the various models we developed along with supporting results for each of them.

3.1 Unigram Model

The corpus we work with has a collection of 200,000 reviews each labeled as positive or negative. As shown in equation in 3 we count the number of time a word appears with a particular label and thus create a distribution over the connotation in which the word is used. For example the word "screwed" usually occurs in negative reviews thus it is likely that when the word "screwed" is encountered in a tweet it ensues a negative connotation.

$$P(Y = 1|w_i) = \frac{C(Y = 1|w_i) + \delta}{C(Y = 1|w_i) + C(Y = 0|w_i) + 2 * \delta} \tag{1}$$

$$\tag{2}$$

$$P(Y = 0|w_i) = \frac{C(Y = 0|w_i) + \delta}{C(Y = 1|w_i) + C(Y = 0|w_i) + 2 * \delta} \tag{3}$$

As the equation 3 shows, we also use smoothing to avoid extreme skews due to hidden biases in the dataset. Apart from the dataset bias, smoothing is a good practice when estimating probabilities from raw counts.

It is interesting to note that we use reviews to create the sentiment language model that will be later used to understand the sentiments of tweets. The hypothesis is that much like reviews are an expression of a person's opinion a tweet is mostly an expression of one's opinion about an entity. Thus, we feel that in the absence of a sentiment wise labeled dataset of tweets, using product reviews is the next best thing.

We use several methods to understand, classify and summarize people's opinions about entities in the dataset using the trained sentiment language model. The staple method is to collect all the tweets associated with a particular entity, treat them as one giant document, and classify the overall sentiment using the learnt model at the end presenting $P(Y = 1|T)$ and $P(Y = 0|T)$ where T is the set of all tweets associated with that entity. Equation 8 has the details. By using γ we ensure that only words with polarizing connotations are considered into the final equation. For example, a word like "suppose" is a non-stop word and its rather unlikely that any decent dataset will show that it has either a strong positive or negative connotation. To overcome the tiny but arbitrary skewness that such words might introduce, we use the γ parameter that will filter words like "suppose".

$$T = \{t_1, t_2, ...\} \tag{4}$$

$$t_1 = \{w_1, w_2, ...\} \tag{5}$$

$$P(Y = 1|T) = \frac{\sum_t^T \sum_w^t 1\{P(Y = 1|w) - P(Y = 0|w) > \gamma\}}{Z} \tag{6}$$

$$P(Y = 0|T) = \frac{\sum_t^T \sum_w^t 1\{P(Y = 0|w) - P(Y = 1|w) > \gamma\}}{Z} \tag{7}$$

$$Z = P(Y = 1|T) + P(Y = 0|T) \tag{8}$$

One obvious caveat of this approach is the dependency on the size of the review dataset. Particularly, it is obvious for humans that some words have a very sharp positive or negative context but if they don't occur enough times in the review dataset their probabilities can be skewed leading to errors at the tweet classification stage.

Table 1 gives examples of a few words and their counts in the reviews dataset. These words are well understood to be positive or negative but their low occurrences in the training set skews them strongly in the other direction.

As a running example to support the analysis of the different sentiment models we choose "Sarah Palin" as the canonical entity. For a quick background, the dataset is from June 2009 and the general sentiment towards Palin was quite negative in the media and from the common man.

3

Table 1: Word Count before priors

Word	Positive Count	Negative Count
hell	1	0
write-off	1	0
facist	3	1
enervating	2	1
connoisseurship	0	1
fuck	1	0
destructiveness	3	1
prostituting	1	0

Table 2 shows the probabilities obtained by using the different sentiment language models.

A deeper analysis of the most frequently occurring words shows that deeply negative words such as shit, fuck, prostiuiting etc. were considering as positive due to their low and incorrect occurrences in the reviews training set. Similar skewed results were also found for other entities.

The next section presents the solution for the above problem.

3.2 Unigram with Prior

After surveying the literature we found that a common technique [3] is to augment the training data with a prior using a hand labeled dataset of words that are explicitly known to be positive or negative. We prepared this dataset by scraping selected sources on the internet.

To incorporate the prior we initialize the counts of the known words with a reasonably larger number. Since these words have been extracted from a hand labeled (or hand parsed) source it is extremely likely that their labeling is accurate. The modified formula for computing probabilities of known words is shown in equation 11. C_p is the prior count we initialize the known word with.

$$P(Y = 1|w_i) = \frac{C(Y = 1|w_i) + C_p(Y = 1|w = i) + \delta}{C(Y = 1|w_i) + C(Y = 0|w_i) + C_p(Y = 1|w = i) + C_p(Y = 0|w = i) + 2 * \delta}$$
(9)

(10)

$$P(Y = 0|w_i) = \frac{C(Y = 0|w_i) + C_p(Y = 0|w = i) + \delta}{C(Y = 1|w_i) + C(Y = 0|w_i) + C_p(Y = 1|w = i) + C_p(Y = 0|w = i) + 2 * \delta}$$
(11)

As table 2 suggests, using the prior led to an improvement in the overall classification of Palin as an entity about whom the opinion is negative. Detailed analysis shows that a large number of words that were previously being incorrectly classified (hell,fuck etc) due to the lack of data were now correctly classified.

Apart from misclassifying infrequently occurring words, the prior can also augment the model by including words that didn't occur in the training set and overriding words that were were indecisive based on their usage in the training set. Example a word like "power" is usually used in a positive sense, but in the case of reviews people can be disgruntled about the amount of a power a product uses there-by resulting in a flatter distribution which will result in it being dropped by our sentiment computation equation 8

Anecdotly, the review set didn't have words such as agree and power, which actually occurred quite frequently in the Palin tweets and thus using the prior helped in better understanding the data.

The use of a single word model even if supporting with a prior is a limiting approach. Often enough, there are words in the English language that are used equally in both positive and negative contexts there-by limiting the accuracy of the analysis. The next section discusses how to overcome this caveat.

Table 2: Sentiment for Sarah Palin

Model	Positive Sentiment	Negative Sentiment
Unigram without Prior	42%	58%
Unigram with Prior	37%	63%
Backoff with Bigram and Unigram	35%	65%

3.3 Bigrams with Backoff

The logical solution to improving performance is to try a higher order model. Along with building a unigram model from the reviews corpus we also build a sentiment language model using bigrams.

The theory is along the lines of the unigram implementation, details are shown in equation 16. The unigram model used in this backoff implementation has been trained with a prior of hand labeled words along with the rest of the 200,000 reviews dataset.

$$P(Y = 1|w_i, w_{i-1}) = \frac{C(Y = 1|w_i, w_{i-1}) + \delta}{C(Y = 1|w_i, w_{i-1}) + C(Y = 0|w_i, w_{i-1}) + 2*\delta} \tag{12}$$

$$\tag{13}$$

$$P(Y = 0|w_i, w_{i-1}) = \frac{C(Y = 0|w_i, w_{i-1}) + \delta}{C(Y = 1|w_i, w_{i-1}) + C(Y = 0|w_i, w_{i-1}) + 2*\delta} \tag{14}$$

$$\tag{15}$$

$$\tag{16}$$

Since we train the sentiment language model on the reviews corpus, we felt that using bigrams alone were not an effective way of computing the sentiment from tweets. Hence, we implemented the backoff model where the initial priority is given to the bigram model however in cases the bigram model doesn't exist we go down to the unigram model to ascertain the probability of the given tweets. Equation 23 presents the details.

$$T = \{t_1, t_2, ...\} \tag{17}$$

$$t_1 = \{w_1, w_2, ...\} \tag{18}$$

$$P(Y = 1|T) = \sum_{t}^{T} \sum_{w_i, w_{i-1}}^{t} \frac{1\{H(w_i, w_{i-1})\}1\{P(Y = 1|w_i, w_{i-1}) - P(Y = 0|w_i, w_{i-1}) > \beta\}}{Z} \tag{19}$$

$$+ \frac{(1 - 1\{H(w_i, w_{i-1})\})1\{P(Y = 1|w_{i-1}) - P(Y = 0|w_{i-1}) > \gamma\}}{Z} \tag{20}$$

$$P(Y = 0|T) = \sum_{t}^{T} \sum_{w_i, w_{i-1}}^{t} \frac{1\{H(w_i, w_{i-1})\}1\{P(Y = 0|w_i, w_{i-1}) - P(Y = 1|w_i, w_{i-1}) > \beta\}}{Z} \tag{21}$$

$$+ \frac{(1 - 1\{H(w_i, w_{i-1})\})1\{P(Y = 0|w_{i-1}) - P(Y = 1|w_{i-1}) > \gamma\}}{Z} \tag{22}$$

$$Z = P(Y = 1|T) + P(Y = 0|T) \tag{23}$$

As with the unigram model, we introduce a β parameter that is tuned such that when evaluating the sentiment only those bigrams are included which have a fairly polarizing effect.

Table 3 presents some interesting bigrams which as stand alone words didn't ensue a clear interpretation. For example, the bigram "lines were" possibly comes from the expression "lines were

Table 3: Bigram Probabilities

Bigram	Positive Probability	Negative Probability
was_shaken	0.13	0.87
against_people	0.31	0.69
found_rather	0.13	0.87
who_missed	0.29	0.71
lines_were	0.35	0.65
recovered_the	0.72	0.28

drawn" which usually has a standoff like negative connotation which would have been non-trivial to understand with a lower order model simply based on the individual words "line" or "were".

The running example for the Palin tweets also shows a further tilt towards the negative and overall the results were much more intuitive and robust with the use of this higher order model.

3.4 Implementational Challenges

It is a worth a note that running the above operations on a set of 60 million tweets is not a trivial task. Each entity must have on ready access a set of all the tweets that were linked to it. Both, the task of collecting tweets for an entity and classifying the sentiment using all those tweets lend themselves to a natural Map-Reduce implementation. Our Ruby implementation, uses a home grown Map-Reduce approach with different processes to split the workload onto the different cores of the quad-core machine used for running these experiments.

4 Visualizing Sentiments

The NLP algorithms that we have presented provide a rich way to sift through millions and millions of opinions to distill the essence of what people's sentiments are about different global entities. The raw numbers, often interesting are not intuitive to facilitate the understanding of a lay person. The probability of an entity's sentiment being positive is theoretically useful but practically not informative.

Thus, in this section we present a set of visualizations rooted in NLP and HCI that make it easier and intuitive to comprehend the results of our work.

4.1 Tweet Classification

To supplement the raw numbers, it is often quite interesting to see some of the most positive and negative opinions. To enable this, instead of classifying all the tweets as one document like in the previous sections we treat each tweet as an individual document and classify it. The math is very similar to equation 8 with the words coming from only one tweet and a slightly different value for γ and β to compensate for the 140 character length of the document.

Below is an example of the top three positive and negative Tweets for the entity Microsoft which was tagged as an organization by the NER. The results section has more examples.

Negative:

```
Great call, Microsoft! Don't make a fundamentally better product,
just throw money at it. Keep it up.
http://is.gd/14toc (via @jeresig)

RT @BillBeavers There's nothing in life so difficult that a
Microsoft manual can't make it completely incomprehensible.
-Douglas Adams

Great propaganda from Microsoft http://bit.ly/cdILv. I
```

Figure 1: Sentiment cloud for entity Brazil

```
understand marketing your product, but some of these are
just flat out lies.
```

Positive

```
Online personal appearance advisor patent - http://bit.ly/MfNca
good to see Microsoft at the forefront of software innovation once more

another Microsoft great contribution to the world RT
@ppk: Well, the Windows Mobile 6.5 browser is a genuine IE6.

very cool branded entertainment series sponsored by Microsoft...
always a winner when bringing Jack Welch to the table http://bit.ly/InyX4
```

4.2   Sentiment Cloud

One of the most intuitive ways to visualize the sentiment for an entity is to see a distribution of terms commonly associated with it in a tag cloud format [15].

Although the NLP side is reasonably accurate it is utopian to expect perfect accuracy or understanding from any such system. Thus, a tag cloud can supplement the raw numbers and overcome minor errors caused by misclassified words.

We classify each word in the tweets associated with an entity and use a function of the word's frequency to map to the font to be used in the tag cloud. Equation 28 presents the details. The function maps the frequency to log space since as described in [15] most of the frequency data is distributed as per the power law and smoothing of the higher frequency items helps smooth the visualization. Figure 1 shows a tag cloud for the entity Brazil.

$$F(f) = \frac{log(f) - log(min)}{log(max) - log(min)} F^* \tag{24}$$

$$F^* = \text{largest font for visualization} \tag{25}$$

$$min = \text{minimum frequency of a token} \tag{26}$$

$$max = \text{maximum frequency of a token} \tag{27}$$

$$\tag{28}$$

4.3   Sentiment Timeline

Often enough, there are events that trigger a flurry of opinions about an entity. Marriages, scandals, movie releases, speeches etc. In such scenarios, global sentiments are not as useful and arguably misleading since they can mask the sentiment in that space of time.
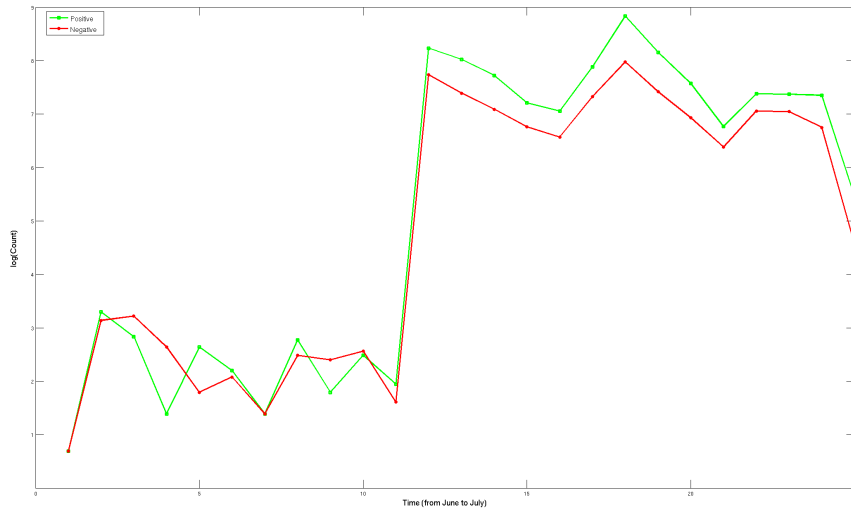
7

Figure 2: Sentiment timeline for Michael Jackson

Table 4: Sentiment comparison for entities

Entity	Positive Sentiment	Negative Sentiment
Internet Explorer	43%	57%
Mozilla Firefox	58%	42%
Microsoft	40%	60%
Apple	55%	45%
Mc Donalds	33%	67%
Starbucks	63%	37%
Stanford	77%	23%
MIT	50%	50%
Madonna	75%	25%
Megan Fox	60%	40%

We provide a day-wise sentiment timeline that shows how the sentiment towards an entity changed over time. The theoretical approach is the same as before. Tweets are grouped based on the day of their creation and the classified sentiment for each day is then plotted on a timeline.

The figure above shows a sentiment timeline for Michael Jackson. The initial entries are from the days of June and show a mixed sentiment leaning mainly towards the negative but the latter entries are from July post the death of Michael Jackson and show an outpouring of positive sentiment.

5 Results

This section showcases some of the results of applying the backoff sentiment language model to the Twitter corpus with 60 million tweets spanning June and July of 2009. All experiments were run on a quad-core machine and the challenges of this large datasets were addressed using a home grown Map-Reduce approach to utilize all the four cores available to us.

Most of the results below should be self-explanatory, though some of them do come with a brief explanation where needed.

The popularity here is simply determined by the volume of tweets and we also present the sentiment associated with each of these "popular" entities.

8

Table 5: Sentiment for Words Events June-July 2009

Entity	Pos	Neg	Top Positive Words	Top Negative Words
Air France	46%	54%	closer, recovered, families	black, hoax, doomed
Fifa	74%	26%	victory, love, entirely	banned, noise, down
Transformers	54%	46%	star, lucky, honor_the	little, out, reviews
Chris Brown	47%	53%	awards, funny, tribute	bitch, ass, beat
Brazil	74%	26%	brilliant, loves, win	upsets, defeated, beat
Bernard Madoff	41%	59%	accord, today, best	fraud, sentenced, disgraced

Table 6: Most Popular Location Entities

Entity	Positive Sentiment	Negative Sentiment
Iran	67%	33%
Tehran	52%	48%
US	57%	43%
UK	62%	38%
London	63%	37%

Table 7: Most Popular Person Entities

Entity	Positive Sentiment	Negative Sentiment
Michael Jackson	64%	36%
Harry Potter	76%	24%
God	67%	33%
Barack Obama	49%	51%
Mir Mousavi	47%	53%

Table 8: Most Popular Organization Entities

Entity	Positive Sentiment	Negative Sentiment
CNN	46%	54%
Back Street Boys	60%	40%
Apple	55%	45%
Reuters	58%	42%
Microsoft	40%	60%

Table 9: Most Popular Username Entities

Entity	Positive Sentiment	Negative Sentiment
tweetmeme (http://twitter.com/tweetmeme)	68%	32%
mashable (http://twitter.com/mashable)	58%	42%
addthis (http://twitter.com/addthis)	67%	33%
peterfacinelli (http://twitter.com/peterfacinelli)	54%	46%
souljaboytellem (http://twitter.com/souljaboytellem)	46%	54%

Figure 3: Sentiment cloud for entity Harry Potter

Figure 4: Sentiment cloud for entity Microsoft

It is interesting to note the impact an external event has on the sentiment timeline. Figure 5 shows the sentiment timeline for entity Sarah Palin. The overall sentiment is quite negative throughout, expect for July 4 when the sentiment is sharply positive. By cross referencing some of the July 4th tweets associated with Sarah Palin we found that she had resigned from her position of Gov. of Alaska on that day, hence instigating a sharp positive reaction from Twitter users.

Table 10: Most Popular URL Entities

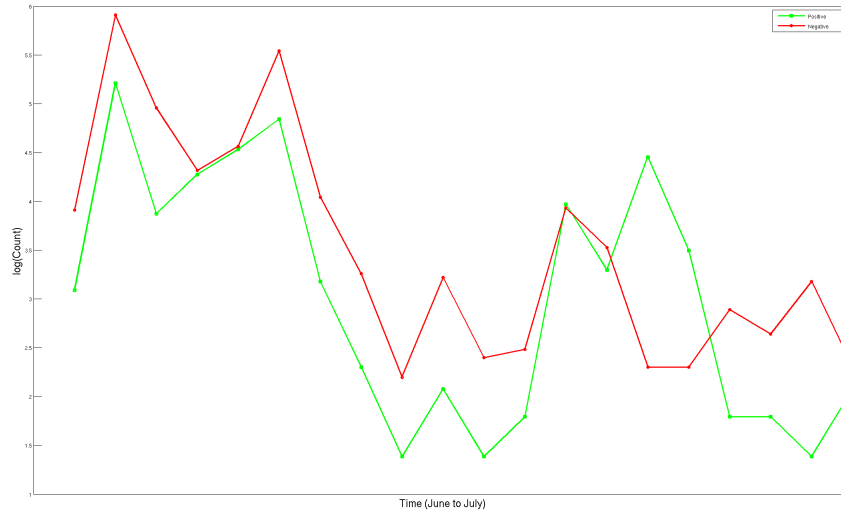| Entity	Positive Sentiment	Negative Sentiment
http://helpiranelection.com/	99%	1%
http://tinyurl.com/	65%	35%
http://wefollow.com/	97%	3%
http://ustre.am/2UhS	46%	54%
http://tweet.sg	51%	49%

Figure 5: Sentiment timeline for Sarah Palin

Top Negative and Positive Tweets for The Entity Iran

Negative:

```
RT f//Iran ..think they made fun of people, don't go here, go this way
not that way & for no apparent reason suddenly attacking random people

RT Iran: I think they made fun of people, don't go here, go this way,
not that way & for no apparent reason suddenly attacking random people

thugs are always stupid RT from Iran "i think they are in my flat now
i am not there they are so stupid thugs #freeiran #Mousavi #Tehran"
```

Positive:

```
For a great perspective on the events in Iran, read Thomas Friedman's
column in the NY times today. It is a good overview of social change.

Let's all send love and blessings to our friends in Iran, and to the planet
as a whole today. Pls RT, as we send prayers of love.

i have gone green in solidarity with Iran who twitter to collaborate in ways
that save their lives and bring freedoms stand with them
```

6 Future Work

The results of our work gave some very interesting insights about the sentiment of global entities. As with any research work, we feel there are a few ways that can improve the overall accuracy. Most useful of these changes will be a Twitter dataset in which tweets have been labeled as positive or negative. Training on reviews is the next best alternative, but the Twitter jargon of 140 characters is challenging to map without a training set also created from Tweets.

Automatic spelling correction during both testing and training can help reduce the arbitrary variance in the model caused by typographical errors in tweets. Stemming can also map the different forms of the same sentiment such as "likes, like, liking" to the same root, there by increasing the sentiment classification accuracy

Perhaps if we had access to a proper Map-Reduce framework such as Amazon Map-Reduce we would run our experiments on a larger set of Twitter data to find more patterns and run more ambitious experiments such as training a semi-supervised sentiment language model by bootstrapping only with a few hundred hand labeled words.

References

[1] Foundations of Statistical Natural Language Processing. C. Manning, H. Schutze. *MIT Press 1999.*

[2] CS229 Machine Learning. A. Ng. *http://www.stanford.edu/class/cs229/, 2010.*

[3] Opinion Summarization of Web Comments. M. Potthast and S. Becker. *Proceedings of the European Conference on Information Retrieval (ECIR), 2010.*

[4] Opinion Cloud for Youtube. B. Stein. *http://www.uni-weimar.de/cms/index.php?id=11254, 2010.*

[5] Twitter. *http://twitter.com/about, 2010.*

[6] Cross-Language Text Classification using Structural Correspondence Learning. P. Prettenhofer and B. Stein. *Proceedings of the Association of Computational Linguistics (ACL'10), 2010.*

[7] Cross-Lingual Sentiment (CLS) dataset. P. Prettenhofer and B. Stein. *http://www.uni-weimar.de/cms/medien/webis/research/corpora/webis-cls-10.html, 2010.*

[8] Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. J.R. Finkel, T. Grenager, and C. Manning. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL), 2005.*

[9] Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. P. Turney. *Proceedings of the Association for Computational Linguistics (ACL), 2002.*

[10] Thumbs up? Sentiment Classification using Machine Learning Techniques. B. Pang, L. Lee and S. Vaithyanathan. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2002.*

[11] Mining and Summarizing Customer Reviews. M. Hu; B. Liu. *Proceedings of KDD, 2004.*

[12] Opinion Observer: Analyzing and Comparing Opinions on the Web. B. Liu, M. Hu and J. Cheng *Proceedings of WWW, 2005.*

[13] Sentiment Analysis and Subjectivity. B. Liu. *Handbook of Natural Language Processing, Second Edition, (editors: N. Indurkhya and F. J. Damerau), 2010.*

[14] Large-Scale Sentiment Analysis for News and Blogs N. Godbole, M. Srinivasaiah and S. Skiena *ICWSM, 2007.*

[15] An Assessment of Tag Presentation Techniques. M. Halvey and M.T. Keane *WWW 2007.*

[16] Stanford University Info Lab. *http://infolab.stanford.edu/, 2010.*