Sentiment Analysis for Twitter using Hybrid Naive Bayes

Harsh Thakkar ¹  Dr. Dhiren Patel ²

¹M.Tech. II Student
²Professor & Guide
Computer Engineering Department
SVNIT, Surat

June 19, 2013
Road Map

1. Introduction
2. Background & Related work
3. Proposed approach
4. Experimental setup
5. Results & Analysis
6. Conclusion
**Sentiment Analysis**

- **Sentiment Analysis**: “It is the phenomenon of extracting sentiments or opinions from reviews expressed by users over a particular subject, area or product online”
Natural Language Processing (NLP): “It is the technology dealing with our most ubiquitous product: human language, as it appears in emails, web pages, tweets, product descriptions, newspaper stories, social media, and scientific articles, in thousands of languages and varieties”
Motivation

Why S.A.?

- Increased use of microblogging as a platform to express opinions.
- Everyday enormous amount of data is created from social networks like twitter.
- Data $\Rightarrow$ Valuable information for everybody’s needs.

Why Twitter?

- Twitter is an Open access social network
- It is an Ocean of sentiments (140 characters High sentiment density)
- Twitter provides developer friendly API mining sentiments is easier
Road Map

1. Introduction
2. Background & Related work
3. Proposed approach
4. Experimental setup
5. Results & Analysis
6. Conclusion
Background & Related work

- Sentiment analysis is formulated as a text-classification problem
- Depending on the task at hand and perspective of the person doing the sentiment analysis, the approach can be:
  - General approaches
  - Twitter specific approaches
General Approaches

- General approaches are as follows:
  - **Knowledge-based approach**: is a $F(x)$ of keywords
  - **Relationship-based approach**: component relationship oriented [customer, brand]
  - **Language models**: is based on frequency of n-grams
  - **Semantics & Discourse structures**: Overall semantic structure of a text is taken into consideration. Every word has its subjective meaning

- Applications:
  - **Movie reviews** [4]
  - **Product reviews** [5]
  - **News and Blogs** ([3],[6])
Twitter specific Approaches

Twitter specific approaches are:

- Lexical approach
- Machine learning approach
- Hybrid approach
Lexical approach

- Is a
- Great
- Recommend
- Etc

Feature extraction

Classification

Formula/e:
* Sentiment of a sentence = \sum \text{score/weight of words.}

* Weight * Feature
  0.03 | is a
  0.25 | great
  0.48 | recommend

Vote:
0.76 \geq \text{some threshold}

\text{POSITIVE!}
Machine learning approach

- **Main tasks:**
  - The classifier (algorithm/method)
  - Selection of features (emoticons, n-grams, etc)
  - The training Data!

- A series of feature vectors are chosen and a collection of tagged corpora are provided for training a classifier.

- Selection of features is crucial to the success rate of the classification.

- Two classification methods are dominant
  - **S.V.M** ([14],[15])
  - **Naive Bayes** [16]
## Performance comparison of Lexical ML approaches

<table>
<thead>
<tr>
<th>Approach</th>
<th>Variants</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lexical [20]</td>
<td>Baseline</td>
<td>50%</td>
</tr>
<tr>
<td>Lexical [20]</td>
<td>Baseline, stemming</td>
<td>50.20%</td>
</tr>
<tr>
<td>Lexical ([8],[10])</td>
<td>Baseline, WordNet</td>
<td>60.40%</td>
</tr>
<tr>
<td>Lexical [20]</td>
<td>Baseline, Yahoo web search</td>
<td>57.70%</td>
</tr>
<tr>
<td>Lexical [20]</td>
<td>Baseline, all above</td>
<td>55.70%</td>
</tr>
<tr>
<td>Machine Learning ([14],[15])</td>
<td>SVM, Unigrams</td>
<td>77%</td>
</tr>
<tr>
<td>Machine Learning ([14],[15])</td>
<td>SVM, Unigrams, Aggregate</td>
<td>65 – 68%</td>
</tr>
<tr>
<td>Machine Learning [16]</td>
<td>Naïve Bayes, Unigrams</td>
<td>75 – 77%</td>
</tr>
<tr>
<td>Machine Learning [16]</td>
<td>Naïve bayes, Unigrams, Aggregate</td>
<td>77-78%</td>
</tr>
</tbody>
</table>
## Performance comparison of Hybrid approaches

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Approach</th>
<th>Variants</th>
</tr>
</thead>
</table>
| 1       | [27]     | - Emoticons-trained classifiers. NB, SVM.  
- Collected texts containing emoticons from Usenet newsgroups. (😊 ; 😞)  
- Claims up to 70% accuracy on test set. |
| 2       | [17]     | - Used single class-two NB classifier & unlabeled data. (+ve , -ve)  
- Claims up to 64% accuracy. |
| 3       | [26]     | - Closest approach  
- Used twitter data to train class-two NB classifier.  
- Claimed 84% accuracy. |
Inference

- It's clear from the results ML approaches are superior to lexical approaches.
- In machine learning approaches, Naive Bayes yield higher accuracy. (IMDB, spam filters)
- Lexical vs Machine Learning ⇒ Time vs Performance
<table>
<thead>
<tr>
<th>Road Map</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong></td>
</tr>
<tr>
<td><strong>2</strong></td>
</tr>
<tr>
<td><strong>3</strong></td>
</tr>
<tr>
<td><strong>4</strong></td>
</tr>
<tr>
<td><strong>5</strong></td>
</tr>
<tr>
<td><strong>6</strong></td>
</tr>
</tbody>
</table>
Problem Statement

“To propose a hybrid approach yearning competitive results by hybridizing machine learning and lexical approaches that captures and analyses sentiments of users in an open social network like twitter for exploring public opinion.”
We propose to hybridize the following two, lexical and machine learning approaches:

- **Lexical** ⇒ SentiWordNet Lexicon dictionary, with;
- **Machine learning** ⇒ Naive Bayes algorithm
Proposed system architecture

Phase I

Corpus

Preprocessing

Sentiment Analysis

Results

Twitter API

Intermediate steps

Phase II

"@alex: This is a tweet, please score a sentiment for it!"

Linguistic resources

SentiWordNet 3.0
Proposed process flow model

Data Gathering
- Twitter API
- Python scripted crawler

Preprocessing
- Lang detect
- Tokenize
- N-grams
- Stop words
- Strip smileys
- Err. casting

Training Data
- Multi-sized ~4 million Tweets Datasets

Classification
- Naïve bayes
- SentiWordNet 3.0

Results
- NLTK
Corpus & Preprocessing

- **Corpus:**
  - We crawled labelled datasets using (😊, 😊) emoticons.
  - It contains various datasets of 1k, 10k, 50k, 100k and 1M tweets, total approx. 4 Million.
  - Data is crawled by archiving realtime tweets via Tweet-Stream API.

- **Preprocessing:**

```
Corpus -> Language Detection -> Tokenize -> N-grams
    |                     |                          |
    v                     v                          v
Erratic Casting        Strip smileys                Stopwords
```

Phase I

- Naive Bayes
- Based on the Bayesian conditional probability model

\[
P(H|E) = \frac{P(H)P(E|H)}{P(E)}
\]  

(1)

where,

- \( P(H|E) \) - posterior probability of the hypothesis.
- \( P(H) \) - prior probability of hypothesis.
- \( P(E) \) - prior probability of evidence.
- \( P(E|H) \) - conditional probability of evidence of given hypothesis.

Or in a simpler form:

\[
Posterior = \frac{(Prior) \times (Likelihood)}{Evidence}
\]  

(2)
Phase II

- Integrating SentiWordNet 3.0:
  - Derived from WordNet (hierarchical organized lexical database)
  - Groups English words into sets of synonyms called “synsets”
  - Records semantic relations between these synonym sets.
  - Each term in SentiWordNet database is assigned a score of $[-1, 1]$ in SentiWordNet which indicates its polarity.

[courtesy:sentiwordnet.isti.cnr.it]
Road Map

1. Introduction
2. Background & Related work
3. Proposed approach
4. Experimental setup
5. Results & Analysis
6. Conclusion
## General system requirements for Hybrid Naive Bayes

<table>
<thead>
<tr>
<th>Component</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>Windows XP/7/8 Linux Ubuntu 12.04</td>
</tr>
<tr>
<td>Processor</td>
<td>C2D/i3/i5/i7 (32 &amp; 64 bit)</td>
</tr>
<tr>
<td>RAM</td>
<td>Min. 4 GB</td>
</tr>
<tr>
<td>Disk storage</td>
<td>Min. 20 GB</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>High speed broadband (uninterrupted)</td>
</tr>
<tr>
<td>Third-party tools &amp; s/w</td>
<td>Linked Media Framework (LMF) 2.3.5, NLTK 2.0, SentiWordNet 3.0</td>
</tr>
</tbody>
</table>
We use the following tools and technologies:

- **Python® 2.7** // Overall scripting & backend
- **SentiWordNet 3.0** // Linguistic resource
- **LMF® 2.3.5** // Persistent data storage
- **NLTK® 2.0** // Language processing and validation
Road Map

1. Introduction
2. Background & Related work
3. Proposed approach
4. Experimental setup
5. Results & Analysis
6. Conclusion
We present the results in the form of classifier accuracy, in two phases:

- **Phase I**: Base naive bayes performance
  - Tests were carried out using multiple twitter datasets consisting of a mixture of new and old keywords e.g. [“ironman3”, “amitabhbachhan”, “google”, “twitter”, “robertdowneyjr”, etc]
  - Results are validated using NLTK 2.0

- **Phase II**: Hybrid naive bayes performance
  - We carried out the same procedure after Integrating SentiWordNet dictionary with the classifier
Phase I: Base naive bayes in action on a windows platform

```python
C:\Python27\test_smm\tracker\collector\trainer>python tweetClassifier.py
Done for n
Done for p
Done training P1
47909 98048
{'tf': 343064, 'df': 0}
Deleting p1 set
Done deleting p1 set
```
Phase I: Accuracy of base naive bayes classifier with a 50k tweets dataset.

```
C:\Python27\test_smm\tracker\tests>python accuracyTest2.py
Most Informative Features
  follow##s## = 1   p : n   =   69.5 : 1.0
  ##s##pleas = 1   p : n   =   48.8 : 1.0
  get##s## = 1   p : n   =   32.7 : 1.0
  ##s##xx = 1   p : n   =   29.3 : 1.0
  love##s## = 1   p : n   =   25.8 : 1.0
  ##s##lol = 1   p : n   =   25.8 : 1.0
  fallback##s## = 1   p : n   =   25.8 : 1.0
  ##s##work = 1   p : n   =   23.5 : 1.0
  alway##s## = 1   p : n   =   23.5 : 1.0
  me##s## = 1   p : n   =   22.4 : 1.0
  ya##s## = 1   p : n   =   22.0 : 1.0
  ##s##follow = 1   p : n   =   18.9 : 1.0
  wish##s## = 1   p : n   =   17.8 : 1.0
  terimakashi = 1   p : n   =   17.8 : 1.0
  ##s##welcom = 1   p : n   =   17.8 : 1.0
  muchthank = 1   p : n   =   16.6 : 1.0
  ##s##thank = 1   p : n   =   15.5 : 1.0
  ##s##well = 1   p : n   =   15.5 : 1.0
  ##s####s## = 1   p : n   =   15.2 : 1.0
  po##s## = 1   p : n   =   14.3 : 1.0
  pleassee = 1   p : n   =   14.3 : 1.0
  thank##s## = 1   p : n   =   13.4 : 1.0
  haha##s## = 1   p : n   =   13.2 : 1.0
  god##s## = 1   p : n   =   13.2 : 1.0
  dong##s## = 1   p : n   =   12.7 : 1.0
None
The accuracy of Classifier in percentage is:
50.7754284164
```
Phase II: Hybrid naive bayes in action on a windows platform

```bash
C:\Python27\Twitter Sentiment Analyzer\analyzer\trainer>python tweetClassifier.py using sentiwordnet dictionary
tweets: 50000  breaks: 0
classified 1000 tweets
classified 2000 tweets
classified 3000 tweets
classified 4000 tweets
classified 5000 tweets
classified 6000 tweets
classified 7000 tweets
classified 8000 tweets
classified 9000 tweets
classified 10000 tweets
classified 11000 tweets
classified 12000 tweets
classified 13000 tweets
classified 14000 tweets
classified 15000 tweets
classified 16000 tweets
classified 17000 tweets
done classify
done training p1
{'en': {'p': 12865, 'n': 4423}}
p2_length: 11206  p1_length: 17288
st: {'tf': 71395, 'df': 0}
deleting p1 set
Done deleting p1 set
```
Phase II: Accuracy of base naive bayes classifier with a 50k tweets dataset.

The accuracy of Classifier in percentage is: 98.5900410494
Phase I: The base naive bayes classifier in action on a Ubuntu 12.04 server with 10k tweets dataset.
Phase I: Accuracy of base naive bayes classifier on a Ubuntu 12.04 server with 10k tweets dataset.
Phase II: Most Informative Features of hybrid naïve bayes approach on 50k Tweet dataset
**Results**

<table>
<thead>
<tr>
<th>Dataset size</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1K</td>
<td>28.54</td>
</tr>
<tr>
<td>10K</td>
<td>29.57</td>
</tr>
<tr>
<td>50K</td>
<td>50.77</td>
</tr>
<tr>
<td>100K</td>
<td>59.96</td>
</tr>
<tr>
<td>1M</td>
<td>70.02</td>
</tr>
</tbody>
</table>

**Base Naïve Bayes performance**

- Accuracy (new)
- Accuracy (old)

<table>
<thead>
<tr>
<th>Dataset size</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1K</td>
<td>63.95</td>
</tr>
<tr>
<td>10K</td>
<td>97.50</td>
</tr>
<tr>
<td>50K</td>
<td>98.60</td>
</tr>
<tr>
<td>100K</td>
<td>95.48</td>
</tr>
<tr>
<td>1M</td>
<td>94.50</td>
</tr>
</tbody>
</table>

**Hybrid Naïve Bayes performance**

- 1K = Thousand
- 1M = Million
Road Map

1. Introduction
2. Background & Related work
3. Proposed approach
4. Experimental setup
5. Results & Analysis
6. Conclusion
We successfully hybridized existing lexical and machine learning approaches and out-performed base nave bayes consistent average accuracy $\geq 90\%$, and $98.6\%$ in the best case.

Our approach also out performs other approaches of ([17], [26], [27]).

It clear from the results that hybrid nave bayes can positively applied over other sentiment analysis applications like

- Financial sentiment analysis (stocks opinion mining)
- Customer feedback services
- etc
Future work

- Interpreting sarcasm
  - “Discourse-driven sentiment analysis”
  - Deep dive into linguistics (Dr. Pushpak & team, IITB)
- Multi-lingual support
  - Language specific lexicon dictionary
Our publication

Harsh Thakkar and Dhiren Patel., "Approaches for sentiment analysis on Twitter: A state-of-art study" accepted at International Network for Social Network Analysis conference (INSNA), Xi’an, China, July 2013.
THANK YOU!

for your patience..

😊 or 😞 ?