Sentiment and Topic Analysis

Presenters:
Abigail Bartolome, Matthew Bock, Rahul Krishnamurthy, Radha Krishnan Vinayagam

CS 6604 TS: Digital Libraries, Spring 2017
Final Team Term Project Presentation
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
Professor Edward A. Fox
Presented May 2, 2017
Acknowledgements

- Dr. Edward A. Fox
- Digital Library Research Laboratory (DLRL)
- Integrated Digital Event Archiving and Library (IDEAL) Grant: IIS-1319578
- Global Event and Trend Archive Research (GETAR) Grant: IIS-1619028
- CS 5604 Topic Analysis Teams of Spring 2016 and Fall 2016
Goal

To build an effective tool that will allow linguists and sociologists to find topics of interest within a collection of tweets and explore the sentiments of the tweets relating to each topic

- Extract and Clean Tweets
- Latent Dirichlet Allocation
- Allow User Interaction
- Sentiment Analysis
- User Interface
Outline

• Workflow
• Preprocessing and Latent Dirichlet Allocation
• Emoji Labeled Sentiment Classification
• Dependency Tree Based Sentiment Classification
• Topic Analysis Interface
• Future Work
Workflow
The planned flow for our system’s tools

1. Tweet Collection
2. Latent Dirichlet Allocation
3. Topics Most Frequently Discussed in Collection
4. Tweets with Topics and Corresponding Probabilities
5. For Each Topic:
   - Sentiment Classification
   - Tweets for Each Topic with Corresponding Sentiment

- Topic A positive: [Tweet 144, Tweet 142, ...]
- Topic A negative: [Tweet 141, Tweet 149, ...]
- Topic B positive: [Tweet 148, Tweet 382, ...]
- Topic B negative: [Tweet 324, Tweet 82, ...]
- Topic C positive: [Tweet 244, Tweet 132, ...]
- Topic C negative: [Tweet 334, Tweet 242, ...]
The planned flow of user interaction with our system’s tools
Preprocessing Tweet Collections and Running LDA
Matthew’s Thesis Framework

- Preprocesses collections of tweet text
  - Handles reading data from source files
  - Processing data into data structures

- Runs LDA
  - Framework provides wrapper for running Spark’s LDA implementation on tweet collections
  - Automatically returns overall topic results and topic tags for each tweet
Data flow within framework
Reading Twitter Data

● Utilize TweetCollectionFactory
  ○ Read data from any supported source
    ■ Simplifies development - easy to change data source
  ○ Creates a collection of Tweet data structures

● Run analysis on TweetCollection data structure
  ○ No need to do any raw text processing manually
  ○ Use provided functionalities to simplify cleaning and pre-processing
Cleaning Twitter Data

- Separate cleaning for LDA and Sentiment Analysis
- Stop words hinder LDA results, but are necessary for our Sentiment Analysis
- Mentions/hashtags hinder sentiment results, but are common topic-defining terms

<table>
<thead>
<tr>
<th>LDA only</th>
<th>Both</th>
<th>Sentiment only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stopwords</td>
<td>URLs</td>
<td>Mentions</td>
</tr>
<tr>
<td>lowercase</td>
<td>Hashtags</td>
<td></td>
</tr>
<tr>
<td>punctuation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RT marker</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LDAWrapper

- Wrapper around Spark’s LDA implementation - works with TweetCollections
- Two sets of results: overall topic results and individual tags for each tweet

**Overall topic results**

<table>
<thead>
<tr>
<th>Topic number 0:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#travel: 0.019037395276840414</td>
<td></td>
</tr>
<tr>
<td>va: 0.017585128058131682</td>
<td></td>
</tr>
<tr>
<td>sunrise: 0.01737546103835172</td>
<td></td>
</tr>
<tr>
<td>catawba: 0.017162258743414125</td>
<td></td>
</tr>
<tr>
<td>halfway: 0.016313580408116864</td>
<td></td>
</tr>
</tbody>
</table>

**Topic number 1:**

| 5: 0.01571722876285292 |                  |
| thruhikers: 0.012579547329638608 |              |
| thruhiker: 0.012566871814079542 |                  |
| thruhike: 0.010651317950181937 |                  |
| thru: 0.010626772087766138 |                  |

**Topic number 2:**

| #taip: 0.026796846204839297 |                  |
| #indigenous: 0.026796846204839297 |              |
| @americanindian8: 0.026146256569298542 |             |
| mcafe: 0.017894562390441637 |                  |
| knob: 0.017894562390441637 |                  |

**Topic tags for each tweet**

<table>
<thead>
<tr>
<th>Result</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability that this tweet belongs to each topic</td>
<td>[0.851, 0.13, 0.019]</td>
</tr>
<tr>
<td>Topic number assigned to this tweet</td>
<td>0</td>
</tr>
<tr>
<td>Topic label assigned to this tweet</td>
<td>“#travel va sunrise catawba halfway”</td>
</tr>
</tbody>
</table>
Emoji-Labeled Sentiment Classification
Emoji Extraction

- UTF-8 format containing non-alphanumeremic codes
  - E.g.,

<table>
<thead>
<tr>
<th></th>
<th>\xf0\x9f\x98\xa0</th>
<th>Angry</th>
<th>Negative</th>
<th>😞</th>
</tr>
</thead>
<tbody>
<tr>
<td>😷</td>
<td>\x0f\x9f\x98\x84</td>
<td>Smiling face with open mouth and smiling eyes</td>
<td>Positive</td>
<td>😊</td>
</tr>
</tbody>
</table>
Sentiment Classifier

- Binomial logistic regression model
- Word2Vec
- Not enough labeled data to train classifier
  - Too many false-negatives
Dependency Tree Based Sentiment Classification
Basics

Lexicon Used:

- VADER (Valence Aware Dictionary and sEntiment Reasoner)
- General Inquirer (polarity reversal words)

A way to compute overall polarity using Lexicon

- Parse Tree created by Syntaxnet

- Our Focus: Impact of polarity reversal words and negation in the overall polarity of tweet
Polarity reversal words with VADER

Tested VADER with tweets that had polarity reversal words.

We focused on very limited words because we were testing the polarity of tweets manually.

We focused on words such as “depression,” “anxiety,” and “stress”

These words have negative sentiment scores.

Then we started looking for tweets which had these negative words along with polarity reversal words like “abate,” “diminish,” “reduce,” and “decrease.”
Polarity reversal words with VADER

Following are subset of tweets that we tested with VADER:

1. "Study shows a significant decrease in depression after taking psilocybin"
   Vader score = -.4404
2. “Escape to nature, even if just for a 30 minute walk.. it will greatly lower your stress levels and reduce risk of depression”
   Vader score = -.309
3. "Singing helps reduce feelings of depression and anxiety, increases oxygen to your lungs."
   Vader score = -.4215
4. “Listening to music for an hour every day can reduce chronic pain by up to 21% and depression by up to 25%”
   Vader score = -.7906
Our Observation on VADER

When we have polarity reversal words in tweets with a negative sentiment word, then the output of VADER is different from the expected value.

This list of tweets is not sufficient to draw any concrete conclusion, and hence we cannot make any claims about the accuracy of VADER.

Our objective of such a test was to find a category of tweets for which it is difficult to predict sentiment.

This led to our search for a method that can determine sentiment of tweets with polarity reversal words.
Parse Tree Based Approach (Rule 1)

Voting with Polarity Reversal

Polarities of nodes in a parse tree are reversed if they have odd numbers of reversal phrases in their ancestors.

Add polarities of all nodes

\[ p = \text{val} \left( \sum_{i=1}^{n} m_i \prod_{j \in A_i} (-1)^{r_j} \right) \]

- \( p = \text{pos, when sum} > 0 \)
- \( p = \text{neg, when sum} < 0 \)
- \( p = \text{neutral, else} \)

\( r_j = 1 \) if there is a reverse polarity word in Ancestor list
Else \( r_j = 0 \)

Set of ancestors
Parse Tree Based Approach (Rule 2)

Reverse polarity of subtrees if head is a polarity reversal word. Add sub trees polarity to get the polarity of overall sentence (Root)
Rule 1 Fails

Both in sentiwordnet and VADER
Score of Negation words in Lexicon is 0.

Tweet -> My Anxiety won’t abate

abate VB root

-0.17

anxiety NN nsubj

won’t RB neg

my PRP poss

Polarity reversal word

Score 0
Rule 2 Fails

Tweet - ->Conversations reduce stigma and increase understanding.

Polarity of increase should not be reversed due to word ‘reduce’

And here ‘and’ is connecting two clauses and not two words.

How to detect that in parse tree?
Improvement

Straightforward application of Rules will give wrong results

Approach 1.

Detection of two independent clauses from a Parse tree.

Compute sentiment on each of them separately and add them.

Example:

Conversation reduces stigma

Conversation increases understanding
Approach 2

Do a sentential analysis. Find subject, verb, and object.

And decide when to reverse polarity based on the head node.

Our Approach:
1. Do further analysis only if the conjunction is connecting two clauses rather than two words.

2. Do not reverse polarity due to a polarity reversal word in the head node, if
   a. The child is a verb and accompanied by a subject in its neighbor
   b. The child has an object as its dependent
3. Use ‘neg’ part of speech to negate the overall polarity of head node.

Verification of Results:

We compared our results with VADER

<table>
<thead>
<tr>
<th>Files after Topic Analysis</th>
<th>Tweet Count</th>
<th>Our Approach Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topics_0</td>
<td>654</td>
<td>97.7%</td>
</tr>
<tr>
<td>Topics_1</td>
<td>552</td>
<td>96.3</td>
</tr>
<tr>
<td>Topics_2</td>
<td>688</td>
<td>99.56</td>
</tr>
</tbody>
</table>
Conclusions

We found limitation in state of art tool VADER

Presented limitations in existing parse tree based approaches

Presented a better rule based approach which overcomes problems in previous approaches

Our approach showed good accuracy for general English tweets and not just tweets related to polarity reversal words.
Topic Analysis Interface (demo)
Number of Topics: 5
Number of Iterations: 100
/home/cloudera/workspace/Sentiment/AT0412.txt

Status
Connecting to Spark...
Connected to Spark

Run
First Pass Results - Reasonable results but not all meaningful
Filtering topic words that don’t contribute meaning
Second Pass Results - Topics start to become more defined
More Filtering
Repeat as Necessary

- Process can be repeated any number of times
- Continue to remove uninteresting terms until topics become meaningful
- Finish button writes result sets to file for sentiment processing
Future Work
Future Work

- Lemmatization and stemming for topic analysis
- More labeled data
  - Incorporate specialized hand-labeled data
  - Use pre-defined dictionary
- Combine NER with parse trees technique to get the polarity of entities
- Use part of speech tags and apply machine learning techniques to determine sentiment of tweet
- Add more granular controls and sentiment analysis to user interface
Questions?