Floods

Joe Acanfora, Myron Su, David Keimig and Marc Evangelista
CS4984: Computation Linguistics
Virginia Tech, Blacksburg
December 10, 2014
Introduction

1. Objective
2. Discussion of corpora
3. Final results
4. Tools we used for cleaning the data
5. Tools we used for language processing
6. Tools we did not use
7. What we learned
8. Conclusion
Objective

Generate summaries of flooding events based on collections of news articles.
Flood Data

- ClassEvent - Islip_Flood
  - 11 Files
- YourSmall - China_Flood
  - 537 files
- YourBig - Pakistan_Flood
  - 20,416 files

Unclean data
In June 2011 a flood spanning 9.94 miles caused by heavy rain affected the Yangtze river in China. The total rainfall was 170.0 millimeters and the total cost of damages was 760 million dollars. The flood killed 255 people, left 87 injured, and approximately 4 million people were affected. In addition 168 people are still missing. The cities of Wuhan Beijing and Lancing were affected most by flooding, in the provinces of Zhejiang Hubei and Hunan. Finally nearly all of the flood damage occurred in the state of China.
In August 2010 a flood spanning 600 miles caused by heavy monsoon affected the Indus River in Pakistan. The total rainfall was 200.0 millimeters and the total cost of damages was 250 million dollars. The flood killed 3000 people, left 809 injured, and approximately 15 million people were affected. In addition 1300 people are still missing.

The cities of Nasirabad, Badheen, and Irvine were affected most by flooding, in the provinces of Sindh, Mandalay, and Punjab. Finally nearly all of the flood damage occurred in the state of Pakistan.
Tools We Used...
1. Removed files less than 5KiB
2. Machine Learning
   a. **DecisionTreeClassifier** = 90%
   b. **NaiveBayesClassifier** = 80%
   c. **MaxEntropyClassifier** = 73%
   d. **SklearnClassifier** = 92%
3. Picked top paragraphs from corpus
   a. Used WordNet on 20 words
   b. Tokenized by paragraph
   c. Picked paragraphs with at least 2 WordNet results
<table>
<thead>
<tr>
<th>Collection</th>
<th>Pre-clean size</th>
<th>Post-clean size</th>
<th>% bytes reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>YourSmall</td>
<td>2.0 MiB</td>
<td>288 KiB</td>
<td>86%</td>
</tr>
<tr>
<td>YourBig</td>
<td>136.7 MiB</td>
<td>3.7 MiB</td>
<td>98%</td>
</tr>
</tbody>
</table>

Merged remaining documents to one for parsing
## Classifier

**Machine learning through decision tree classifier**

<table>
<thead>
<tr>
<th></th>
<th>Accurate</th>
<th>Inaccurate</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>YourSmall</td>
<td>90</td>
<td>10</td>
<td>90%</td>
</tr>
<tr>
<td>YourBig</td>
<td>83</td>
<td>17</td>
<td>83%</td>
</tr>
</tbody>
</table>
Frequency Analysis

- Purposes
  - Cleaning data
  - Generating summary
  - Building YourWord list
Used the POS tagger for our regular expression “cause” string

Checked to see if the cause string returned by the regular expression contained some subject (noun)

In June 2011 a flood spanning 9.94 miles caused by heavy rain affected the Yangtze River in China.
- Best used on cleaned data
  - Patterns prevalent in news reports
  - Same methods of describing flooding event
Regex examples

- "affected by _____", "result of _____", "caused by _____", "by _____"
- day/month/year
- _____ people killed/missing/injured
- _____ (b|m|tr|etc...)illions dollars
- _____ miles/km/etc...
Rather than using the NER tagger for tagging locations we decided to use a Google Maps API...
Contextualizing Locations

- Google Geocoder API
- pygeocoder Python package
Tools We Did Not Use...
Not used extensively
- Bigrams were good, but already in YourWords
- Operations we used were based on single words
- Did help with regex
<table>
<thead>
<tr>
<th>Useful bigrams</th>
<th>YourWords</th>
</tr>
</thead>
<tbody>
<tr>
<td>flash flooding</td>
<td>flood</td>
</tr>
<tr>
<td>heavy rains</td>
<td>rain</td>
</tr>
<tr>
<td>inches rain</td>
<td>overflow</td>
</tr>
<tr>
<td>rain fell</td>
<td>dam</td>
</tr>
<tr>
<td></td>
<td>storm</td>
</tr>
<tr>
<td></td>
<td>severe</td>
</tr>
<tr>
<td></td>
<td>water</td>
</tr>
<tr>
<td></td>
<td>damage</td>
</tr>
<tr>
<td></td>
<td>submerge</td>
</tr>
<tr>
<td></td>
<td>washed</td>
</tr>
<tr>
<td></td>
<td>collapsed</td>
</tr>
<tr>
<td></td>
<td>river</td>
</tr>
<tr>
<td></td>
<td>discharge</td>
</tr>
<tr>
<td></td>
<td>downpour</td>
</tr>
<tr>
<td>flood</td>
<td>flash</td>
</tr>
<tr>
<td>torrential</td>
<td>sweep</td>
</tr>
<tr>
<td>runoff</td>
<td></td>
</tr>
<tr>
<td>Useful bigrams</td>
<td>Some regexes</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>flash flooding</td>
<td>`(\d+.\d+\smillimeters)</td>
</tr>
<tr>
<td>heavy rains</td>
<td><code>due\sto(\s[A-Za-z]{3,})\{1,3}\result\sof(\s[A-Za-z]{3,})\{1,3}\caused\sby(\s[A-Za-z]{3,})\{1,3}\by\s([A-Za-z]{4,})\{1,2}\heavy\s([A-Z a-z]{3,})</code></td>
</tr>
<tr>
<td>inches rain</td>
<td></td>
</tr>
<tr>
<td>rain fell</td>
<td></td>
</tr>
</tbody>
</table>
Clustering & Mahout

- Documents similar enough that clusters would be indistinguishable
- Wanted data from all good sources
- Clean data was good enough
- Finds multitoken sequences
- Knowledge of existing data
  - brainstormed our own chunks, which was good enough
  - would be helpful if we didn’t know patterns
- Regular expressions alone did the job well on clean data
Conclusion
Wrap Up - Challenges

- New Technologies
  - Hadoop - Map/Reduce
  - NLTK Library
- Group Logistics
  - Times
  - Work Distribution
Wrap Up - Strengths

- Technical Strengths
  - Python
  - LaTeX

- Team Strengths
  - Willing to learn
  - Team synergy
Conclusion - Improvements

- Underestimates
  - Deaths
  - Damages
  - Build statistical model to improve accuracy

- Spatial locations
  - Mean distances
  - Generate map using Google API
Citations

https://pypi.python.org/pypi/geocoder/0.9.1
http://www.nltk.org/book_1ed
Many Thanks

Dr. Edward Fox
GTA Tarek Kanan
GTA Xuan Zhang
GRA Mohamed Magdy Gharib Farag

National Science Foundation, Computing in Context, NSF
DUE-1141209
Villanova