Evaluating Sentiment Analysis Tools Against Developer Commit Logs

Zachary J. Szewczyk
Capstone Presentation
Computer Science and Information Systems Department

December 12, 2017

Background

- Started web design at 12
- CSIS Student 7 years later
- Army ROTC Cadet now
- Commissioning into the Army Cyber Division in the Spring

Motivation

- Lead, plan and direct both defensive and offensive cyberspace maneuvers and effect operations in cyberspace domain
- Conduct OCO by using cyber capabilities in cyberspace to target and neutralize threats
- Conduct DCO to protect data, networks, net-centric capabilities, and other systems through detection, identification, and response to attacks on friendly networks
- **Execute mission command of cyber maneuver forces during DCO and OCO missions in support of joint and combined arms operations**

Sentiment analysis

the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc., is positive, negative, or neutral.

- Oxford

Research Goals

1. Compare the performance accuracy of three sentiment analysis tools against a gold standard, a human's analysis of the sentiment present in a subset of the commits processed by the algorithms

2. Provide concrete examples of inconsistencies across the three sentiment analysis tools

Tools Used

- Mining
    - Boa Infrastructure (http://boa.cs.iastate.edu/)
- Sentiment analysis algorithms
    - SentiStrength (http://sentistrength.wlv.ac.uk/)
    - CoreNLP (https://stanfordnlp.github.io/CoreNLP/)
    - SentiCR (https://github.com/senticr/SentiCR)

Dataset

41,338

The number of commits in the dataset, captured using Boa.

```
foreach (i: int; def(p.code_repositories[i]))
    foreach (j: int; def(p.code_repositories[i].revisions[j]))
        if (!match("empty log message", p.code_repositories[i].revisions[j].log))
            commits[p.id] << p.code_repositories[i].revisions[j].log;
```

Methodology

Processing raw data into a form compatible with each algorithm

commits[102990] = Fixed major problem where
↪️WidgetManager was scoped globally
instead of per-session.

Added support for binding and looking up ServletContext,
↪️HttpSession,
HttpServletRequest & HttpServletResponse from the
↪️context of the current
thread.

commits[102990] = * Moved datasource loading code to
↪️TransactionManager.
* Ensured that datasources are persistent

Methodology

Processing raw data into a form compatible with each algorithm

Fixed major problem where WidgetManager was scoped globally instead of per-session. Added support for binding and looking up ServletContext, HttpSession, HttpServletRequest & HttpServletResponse from the context of the current thread.

Moved datasource loading code to TransactionManager. Ensured that datasources are persistent.

Methodology

Processing raw data into a form compatible with each algorithm

```
# Function to strip special characters from beginning of each line
def sanitize(line):
    # Remove whitespace, ex. from indentation
    line = line.strip()
    # Remove bullets from bulleted lists
    line = re.sub("^[\*-]\s*", "", line)
    # Remove item identifiers from alpha-numeric lists
    line = re.sub("^[A-Za-z0-9]\)\s+", "", line)
    return line

...

# Output bookkeeping data
## Confirm with regex: commits\[[0-9]+
print "Number of commits processed: %d" % commit_count
# Confirm with regex: commits\[[0-9]+\]\s=\s+$
print "Number of blank commits: %d" % blank_commits
print "Number of commits algorithm should process: %d" %
    (commit_count-blank_commits)
```

Scripts

- Three similar scripts, one for each algorithm
  - Given an input file, fed the commit messages to the algorithm
  - Processed output file
  - Aggregated sentiment analysis results for each algorithm

```
for line in fd:
    line = line.strip()
    line = re.sub("\t", ",", line)
    sentiment = int(line.split(",")[0])
    print "SENTIMENT: %d" % sentiment
    print line
    if (sentiment == 0):
        neutral += 1
    elif (sentiment == 1):
        positive += 1
    elif (sentiment == -1):
        negative += 1

print """\
Pos   Neg   Neu
%-6i%-6i%-6i
""" % (positive, negative, neutral)
```

Tool, SentiStrength

SentiStrength		
-1	Fixed problem of wrong resource key being passed.	Fixed problem[-2] of wrong[-2] resource key being passed .[sentence: 1,-2] [result: max + and - of any sentence][overall result = -1 as pos<-neg]

Tools, Stanford CoreNLP

<table>
<tr>
<td>Stanford CoreNLP</td>
<td>
<pre>
<sentence id="17" line="17" sentimentValue="1"
sentiment="Negative">
    <tokens>
        <token id="1">
            <word>Fixed</word>
            <lemma>fix</lemma>

            <CharacterOffsetBegin>787</CharacterOffsetBegin>
            <CharacterOffsetEnd>792</CharacterOffsetEnd>
            <POS>VBN</POS>
            <NER>O</NER>
            <sentiment>Neutral</sentiment>
          </token>
          ...
      </tokens>
      ...
</sentence>
</pre>
</td>
<td>Negative</td>
</tr>
</table>

Tool, SentiCR

SentiCR
Neutral

More Scripts

- Comparison script
  - Find sentence categorization in CoreNLP
  - Compared to SentiStrength and SentiCR
  - Output divergent categorizations

```
for line in c:
    if (re.search("sentimentValue", line)):
        core_nlp = 
line.split("sentiment=")[1].replace("\"", 
"").replace(">", "").replace("Verynegative", 
"Negative").replace("Verypositive", "Positive").strip()
        # print core_nlp
        s_line = s_fd.readline()
        senticr_line = senticr_fd.readline()
        temp = int(s_line.split("\t")[0])
        if (temp == -1):
            sentistrength = "Negative"
        elif (temp == 0):
            sentistrength = "Neutral"
        elif (temp == 1):
            sentistrength = "Positive"
        if not (core_nlp == sentistrength == 
senticr_line):
            i += 1
            print "\""+s_line.split("\t")[1].strip()+"\""
            print " -- Stanford CoreNLP rates this as %s, 
SentiStrength rates this as %s, SentiCR rates this as 
%s." % (core_nlp, sentistrength, senticr_line)
            print
```

Results by Tool

Tool	Positive	Negative	Neutral
SentiStrength	3516	5827	31994
Stanford CoreNLP	3840	17408	19770
SentiCR	90	189	41059

Tool	Positive	Negative	Neutral
SentiStrength	8.51%	14.10%	77.40%
Stanford CoreNLP	9.30%	42.87%	47.83%
SentiCR	0.22%	0.46%	99.33%

Results by Human

Representation	**Positive**	**Negative**	**Neutral**	**Total**
Raw	122	38	840	**1000**
Percentage	12.20%	3.80%	84.00%	**100%**

Results Compared

Method	Positive	Negative	Neutral
Human	12.20%	3.80%	84.00%
SentiStrength	8.51%	14.10%	77.40%
Stanford CoreNLP	9.30%	42.87%	47.83%
SentiCR	0.22%	0.46%	99.33%

Discussion

- Why the disparity?
- Inordinate emphasis on certain words
  - Stanford CoreNLP considers "error", "remove", "compensate", and "change" negative
- Lack of context
  - “Changed embedded tomcat to full tomcat.” and “Renamed BeanFactory->BeanManager.”, both of which Stanford CoreNLP categorized as negative, are not.

Method	Positive	Negative	Neutral
Human	12.20%	3.80%	84.00%
SentiStrength	8.51%	14.10%	77.40%
Stanford CoreNLP	9.30%	42.87%	47.83%
SentiCR	0.22%	0.46%	99.33%

Discussion, continued

- None of the sentiment analysis algorithms agreed for this commit
- Correct sentiment present in this message is neutral

"Returned back to FlowInput / FlowOutput approach, reinstated support."

-- CoreNLP rates this as Negative, SentiStrength rates this as Positive, SentiCR rates this as Neutral.

Discussion, continued

- All three algorithms agreed: this commit message is positive

"Added a cool touchgraph view. Need to spruce it up a bit though."

-- CoreNLP rates this as Positive, SentiStrength rates this as Positive, SentiCR rates this as Positive.

Discussion, continued

- All three algorithms agree that this commit message is negative
    - Presence of a single key word: "fail"
- Correct sentiment present in this message is neutral

"tests ... changed folder-structure ... some tests fail at the moment!"

-- CoreNLP rates this as Negative, SentiStrength rates this as Negative, SentiCR rates this as Negative.

Discussion, continued

- All three algorithms agree that this commit message is neutral

"Renamed old model."

-- CoreNLP rates this as Neutral, SentiStrength rates this as Neutral, SentiCR rates this as Neutral.

Related Work

- D. K. Ly, K. Sugiyama, Z. Lin, and M.-Y. Kan, “Product review summarization from a deeper perspective,” in *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, ser. JCDL ’11. New York, NY, USA: ACM, 2011, pp. 311–314. [Online]. Available: http://doi.acm.org/10.1145/1998076.1998134

- R. Jongeling, P. Sarkar, S. Datta, and A. Serebrenik, “On negative results when using sentiment analysis tools for software engineering research,” *Empirical Softw. Engg.*, vol. 22, no. 5, pp. 2543–2584, Oct. 2017. [Online]. Available: https://doi.org/10.1007/s10664-016-9493-x

Conclusion & Further Work

- Go-to algorithms are ill-suited to the software engineering domain
- Further work or better training is needed to develop an appropriate algorithm
- As of today, SentiCR is not that tool

Tools Used

- Mining
  - Boa Infrastructure (http://boa.cs.iastate.edu/)
- Sentiment analysis algorithms
  - SentiStrength (http://sentistrength.wlv.ac.uk/)
  - CoreNLP (https://stanfordnlp.github.io/CoreNLP/)
  - SentiCR (https://github.com/senticr/SentiCR)

Dataset

41,338

The number of commits in the dataset, captured using Boa.

```
foreach (i: int; def(p.code_repositories[i]))
  foreach (j: int; def(p.code_repositories[i].revisions[j]))
    if (!match("empty log message", p.code_repositories[i].revisions[j].log))
      commits[p.id] << p.code_repositories[i].revisions[j].log;
```

Results Compared

Method	Positive	Negative	Neutral
Human	12.20%	3.80%	84.00%
SentiStrength	8.51%	14.10%	77.40%
Stanford CoreNLP	9.30%	42.87%	47.83%
SentiCR	0.22%	0.46%	99.33%

Questions?