An Overview of Word Embedding-based Machine Learning Methods in Topic Recognition Tasks

Sarah Wiegreffe
Advised by Dr. Paul Anderson
Dept. of Computer Science
Why Natural Language Processing (NLP)?

Lots of Data

Computer Understanding

Topic Recognition

Sentiment Analysis
What is machine learning?

http://sebastianraschka.com/Articles/2014_intro_supervised_learning.html
Traditional machine learning algorithms need fixed-length feature vectors.

- How to generate?
- Traditionally done by Bag of Words

The dog is on the table.
Word Embeddings!

- **Word2Vec**
  - *Distributed Representations of Words and Phrases and their Compositionality*, Mikolov et. al.
  - Fixes word similarity problem

```
“King” - “man” + “woman” = “Queen”
```
Word2Vec Inversion

- Combining preprocessing and machine learning steps
- Neural network architecture

- Document Classification by Inversion of Distributed Language Representations by Taddy.
"This movie is like the thousand "cat and mouse" movies that preceded it. (The following may look like a spoiler, but it really just describes a large class of movies) There is the passionate, wise main character, his goofy but well-meaning sidekick with his ill-placed attempts at humorous comments, the initially-hostile but soon softened gorgeous lady who triggers the inevitable "unlikely" love story, the loved ones taken hostage, and of course the careless evil adversary with his brutal minions. Everybody has seen tons of these movies already, and "National Treasure" is like any one of them, with only a slightly modified wrapping. Every turn of the story was easily predicted (and I can assure you I am not the sharpest tool in the shed). I am quite tired of feeling tricked for money after exiting the theater from a Hollywood movie, and if you have ever felt that way too, heed my warning; stay miles away from this movie."

Sentence Scores:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1 doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5986</td>
<td>0.4014</td>
</tr>
<tr>
<td>1</td>
<td>0.00792023</td>
<td>0.99208</td>
</tr>
<tr>
<td>2</td>
<td>0.369248</td>
<td>0.630752</td>
</tr>
<tr>
<td>3</td>
<td>0.445318</td>
<td>0.554682</td>
</tr>
<tr>
<td>4</td>
<td>0.999581</td>
<td>0.000419095</td>
</tr>
</tbody>
</table>
Hypothesis:

Instead of averaging together sentence scores, can allowing a computer to learn what the most important features of a review or positions of sentences are improve prediction accuracy?

- We strive to do this with as little human input as possible for generalizability.
Our Methods- employing meta-features and ensemble learning

Each sentence of a review receives a probability score from the Word2Vec Inversion algorithm.

Mean of scores
Minimum score
Maximum score
Standard Deviation of scores
Score of first sentence in review
Score of last sentence in review
Number of sentences in review
Sigmoidal function (k=10)
Sigmoidal function (k=20)
Position of min. score in review
Position of max. score in review

Decision Tree or Random Forest Classifier

The generated metafeatures become a fixed-length feature vector used to train either a decision tree or a random forest classifier.
Datasets

- IMDb movie reviews (100,000)
  - Positive or negative sentiment?
- Amazon Fine Food reviews
  - 1-5 star ratings
## Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Bag of Words + Random Forest</th>
<th>Word2Vec Inversion</th>
<th>Inversion + Decision Tree of Meta-features</th>
<th>Inversion + Random Forest of Meta-features</th>
<th>Bagging</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDb movie reviews</td>
<td>83.97%</td>
<td>87.71%</td>
<td>87.98%</td>
<td>88.23%</td>
<td>79.97%</td>
</tr>
<tr>
<td>Amazon Fine Food reviews</td>
<td>70.49%</td>
<td>73.17%</td>
<td>71.03%</td>
<td>75.47%</td>
<td></td>
</tr>
</tbody>
</table>
Next Steps

- Build confidence intervals to test more robustly
  - To account for randomness in Word2Vec model construction
- Rerun all models on a third dataset
  - Twitter data
- Submit for publication in May
Summary/Overview

● Word2Vec Inversion methods show slight improvement over current methods.
● With more robust testing, we hope to discern if they can beat baseline models more generally.
● If so, this is a great improvement not only for achieving higher accuracy of classification, but also for simplifying the process of building machine learning classifiers on text.
Tools

```python
import json
import pandas as pd
import numpy as np
import pickle

#JSON for Python

with open("sample.json", 'r') as f:
    the_data = [json.loads(l) for l in f]

type(the_data[0])

# for smaller file

medicarePUF1 = pd.read_table('./Big_Files/Medicare_Provider_Util_Payment_PUF_CY2014.txt', skiprows = [1])

# have mixed types. Specify dtyp option on import or set low_memory=False.

#interactivity-interactivity, compiler-compiler, result-result

def PUFtoJSON():
    PUF = {}
    cols = medicarePUF1.columns.tolist()[16:]
    for col in cols:
        PUF[col] = str(medicarePUF1[col][j])
    return PUF

print(PUF)
```
Thank you to Dr. Anderson, the Data Science program, the Computer Science department, the Honors College, and Dr. Jihad Obeid in the MUSC Biomedical informatics Center.