Trump + President = Shakedown

Measuring Ideological Differences with Word Embeddings

Mike Burnham
mlb6496@psu.edu
github.com/MLBurnham/word_embeddings

December 11, 2019

Penn State University

Table of contents

Introduction

Background

Two dominant methods in this space:

1. Dictionary methods
   - Inherently noisy
2. Sentiment analysis
   - Lacks context

Word embeddings were invented in 2013 and alleviate both of these problems. However, their use has almost exclusively been limited to predictive rather than explanatory applications

Background and Lit Review

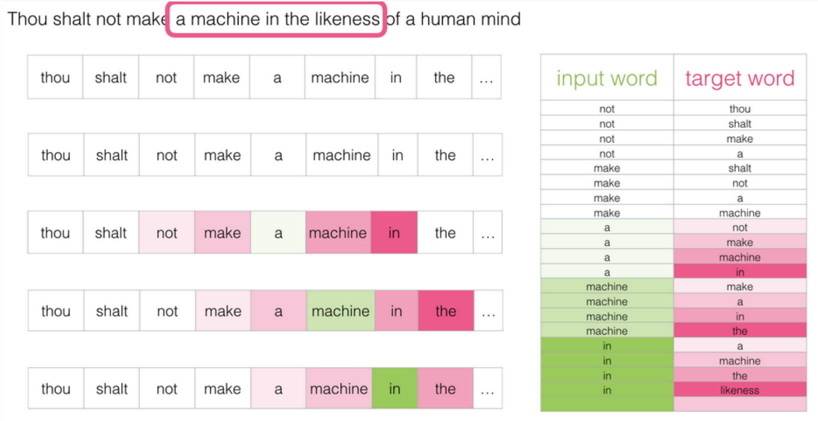
What is a word embedding?

Word embeddings are language models that represent words as vectors of numbers. Those vectors are defined by the other words surrounding that word. Two important implications for this research:

1. This allows us to do math with words
2. Word vectors are not universal

Distributional Hypothesis: Words that appear in the same context tend to have similar meaning

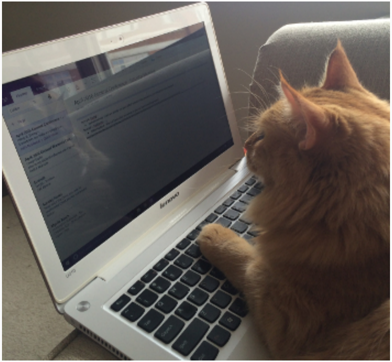
What is a word embedding? Example



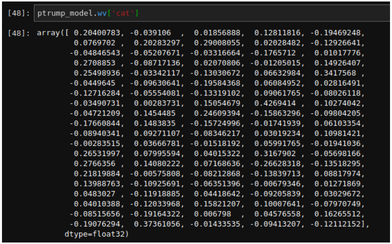
input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine
a	not
a	make
a	machine
a	in
machine	make
machine	a
machine	in
machine	the
in	a
in	machine
in	the
in	likeness

[1]

What is a word embedding? Example



=



```
[48]: ptrump_model.wv['cat']

[48]: array([ 0.20400783, -0.039106  ,  0.01856888,  0.12811816, -0.19469248,
         0.0769702 ,  0.20283297,  0.29008055,  0.02028482, -0.12926641,
        -0.04846543, -0.05207671, -0.03316664, -0.1765712 ,  0.01017776,
         0.2708853 , -0.08717136,  0.02070806, -0.01205015,  0.14926407,
         0.25498936, -0.03342117, -0.13030672,  0.06632984,  0.3417568 ,
        -0.0449645 , -0.09630641, -0.19584368,  0.06084952,  0.02816491,
        -0.12716284, -0.05554081, -0.13319102,  0.09061765, -0.08026118,
        -0.03490731,  0.00283731,  0.15054679,  0.4269414 ,  0.10274042,
        -0.04721209,  0.1454485 ,  0.24609394, -0.15863296, -0.09804205,
        -0.17660844,  0.1483835 , -0.15724996, -0.01741939,  0.06103354,
        -0.08940341,  0.09271107, -0.08346217,  0.03019234,  0.10981421,
        -0.00283515,  0.03666781, -0.01518192,  0.05991765, -0.01941036,
         0.26531997,  0.07995594,  0.04015322,  0.3167902 , -0.05698166,
         0.2766356 ,  0.14080222,  0.07168636, -0.26628318, -0.13518295,
         0.21819884, -0.00575808, -0.08212868, -0.13839713,  0.08817974,
         0.13988763, -0.10925691, -0.06351396, -0.00679346,  0.01271869,
         0.0483027 , -0.11918885,  0.04418642, -0.09205839,  0.03029672,
         0.04010388, -0.12033968,  0.15821207,  0.10007641, -0.07970749,
        -0.08515656, -0.19164322,  0.006798  ,  0.04576558,  0.16265512,
        -0.19076294,  0.37361056, -0.01433535, -0.09413207, -0.12112152],
      dtype=float32)
```

What is a word embedding? Example

```
In [25]: model.similar_by_vector(model['Obama'] - model['USA'] + model['France'])
Out[25]: [('Sarkozy', 0.704483687877655),
          ('Obama', 0.7015002369880676),
          ('President_Nicolas_Sarkozy', 0.642586350440979),
```

```
In [15]: model.similar_by_vector(model['Dublin'] - model['Ireland'] + model['France'])
Out[15]: [('Paris', 0.740702748298645),
          ('France', 0.6797398924827576),
          ('Issy_les_Moulineaux', 0.6246635317802429),
```

[1]

What is a word embedding? Example

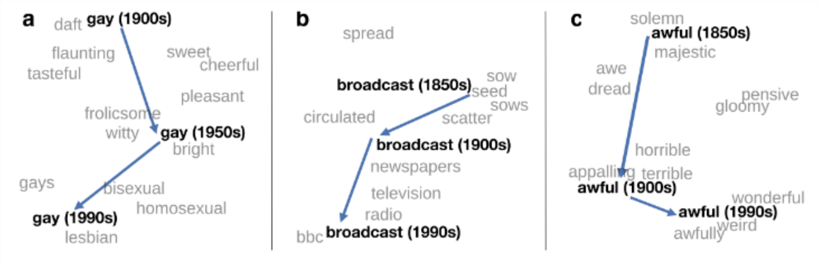
```
In [25]: model.similar_by_vector(model['Obama'] - model['USA'] + model['France'])

Out[25]: [('Sarkozy', 0.704483687877655),
          ('Obama', 0.7015002369880676),
          ('President_Nicolas_Sarkozy', 0.642586350440979),
```

```
In [15]: model.similar_by_vector(model['Dublin'] - model['Ireland'] + model['France'])

Out[15]: [('Paris', 0.740702748298645),
          ('France', 0.6797398924827576),
          ('Issy_les_Moulineaux', 0.6246635317802429),
```

[1]

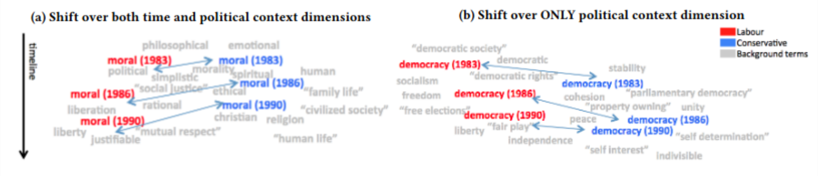
```
[45]: ptrump_model.wv.most_similar(positive = ['trump', 'president'], negative = [], topn = 2)

[45]: [('trumps', 0.8091122508049011), ('shakedown', 0.7698360085487366)]
```

Literature: Diachronic Word Embeddings



[2]

Literature: Words are Malleable



[3]

Questions

1. Can word embeddings capture the qualitative dimensions of political positions?
2. Can word embeddings capture the quantitative dimensions of political positions?
3. If so, what is the most effective implementation?

Questions

1. Can word embeddings capture the qualitative dimensions of political positions? **Yes**
2. Can word embeddings capture the quantitative dimensions of political positions?
3. If so, what is the most effective implementation?

Questions

1. Can word embeddings capture the qualitative dimensions of political positions? **Yes**
2. Can word embeddings capture the quantitative dimensions of political positions? **Maybe**
3. If so, what is the most effective implementation?

Questions

1. Can word embeddings capture the qualitative dimensions of political positions? **Yes**
2. Can word embeddings capture the quantitative dimensions of political positions? **Maybe**
3. If so, what is the most effective implementation? **Unknown**

This research re-examines the first two questions, and explores a new implementation by comparing word vectors within a single vector space, rather than across vector spaces.

Research Design

Hypotheses

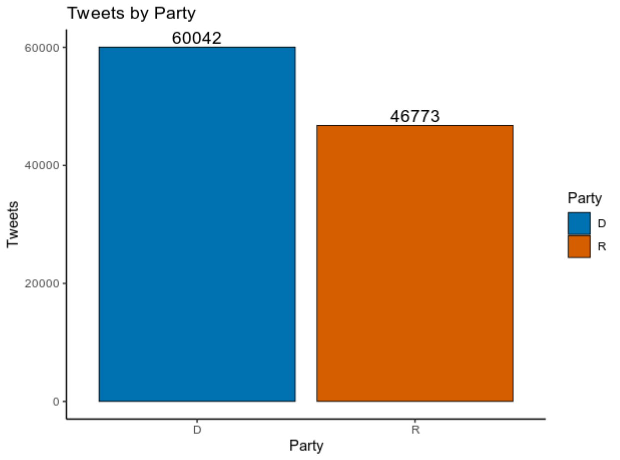
1. A word that represents a political division between democrats and republicans will have a lower cosine similarity with its counterpart than a word that represents agreement.
2. The words in closest proximity to divisive words will highlight points of contention between groups.

Process

1. Collect Data
2. Identify key words and their synonyms
3. Pre-process text
4. Train embeddings
5. Analyze results

1. Collect Data

- Data should have a clear division between ideological groups and present clear ideological stances
- All tweets from Democratic or Republican Congress members between 11/06 and 12/10



2. Identify Keywords

There types of keywords: disagree, agree, and baseline.

Good keywords have three qualities:

1. Represent a clear point of (dis)agreement with well formulated positions.
2. Have a high usage
3. Have an unambiguous interpretation and narrow context

E.g. Impeachment is a good keyword, healthcare is not.

Label	n	Examples
Disagree	30	trump, abortion, gun
Agree	8	veteran, isis, infrastructure
Base	53	answer, opportunity, member

Table 1: Key Words

3. Pre-process text

- Remove punctuation, emoji, digits. Convert everything to lower case.
- Remove stopwords based on a modified version of spaCy's stopwords list
- Isolate and tag key word of interest
- Lemmatize text using spaCy's small english language model

Train embeddings

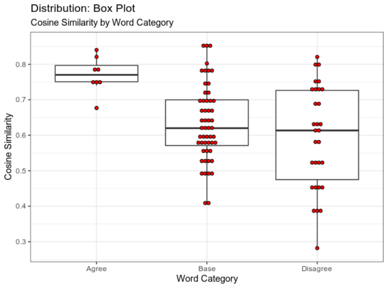
- A new embedding is trained for each key word
- Used Gensim's Word2vec implementation
    - skip-gram architecture
    - 100 dimensions
    - window size = 10
    - learning rate = .025

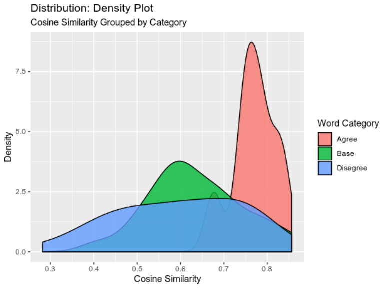
Results

Distribution

Table 2: Cosine similarity distribution

Label	Mean	Min	Max	n
Disagree	0.59	0.28	0.82	30
Agree	0.77	0.68	0.84	8
Base	0.63	0.41	0.86	53





Significance Tests: All Words

	word	similarity	label
1	abortion	0.7978410	Disagree
2	administration	0.4589447	Disagree
3	border	0.5736459	Disagree
4	conservative	0.6800396	Disagree

Table 3: Difference of means relative to disagree words

Label	Mean	n	t-test (p)	Permutation (p)	Bootstrap (CI)
Disagree	0.59	30	-	-	-
Agree	0.77	8	4.27e-06	.008	(-0.24, -0.11)
Base	0.63	53	0.19	0.18	(-0.1, 0.02)

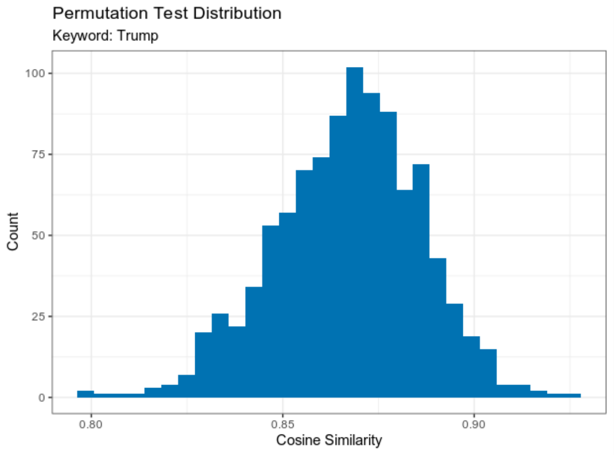
Significance Tests: Trump

Observed value: 0.38

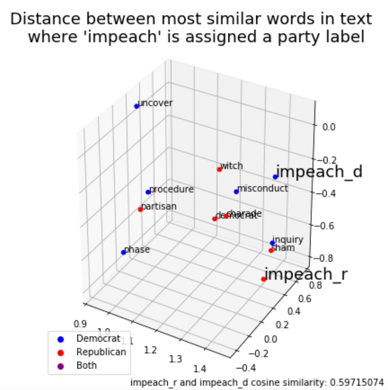
Mean permutation value: 0.87

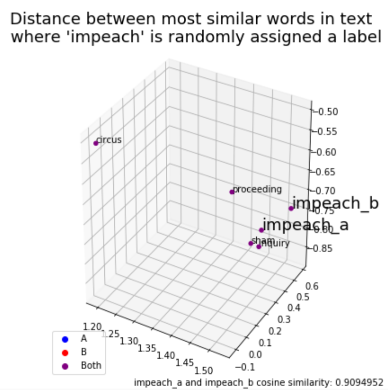
Min permutation value: 0.80

p-value: 0.0



Qualitative Tests: Impeach





Conclusion

Next Steps

- Compare results with alternative methods, (Procrustes, fightin words)
- Test on a larger and more suitable data set (troll tweets?)
- Introduce new dimensions for capturing disagreement (sentiment)
- If successful, build a generalizable pipeline

Bibliography

- Jay Alammar *The Illustrated Word2vec* http://jalammar.github.io/illustrated-word2vec/
- William L. Hamilton, Jure Leskovec, Dan Jurafsky *Diachronic Word Embeddings Reveal Statistical Laws ofSemantic Change* arXiv preprint arXiv:1605.09096 (2016).
- Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, Jaap Kamps *Words are Malleable: Computing Semantic Shifts in Political and Media Discourse* In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1509-1518. ACM, 2017.