Learning Word Representations

Vivek Kulkarni
Understanding Textual Content

- Computationally analyze textual content to understand text

- Dominant approach to analyzing/understanding text is Statistical Learning
  - Learn the appropriate input to output transformation from data!
  - Pro: No need to laboriously design complex rule based systems
  - Better Generalization
The learning from data paradigm

Data → Features (Representation) → Model → Is it a cat?
Representing Text

- How to represent text?
- Choose what granularity is used for representation
  - Document
  - Sentence/Phrases
  - Words
  - Characters
- Properties of a good representation
  - Useful for the task
  - Allow the model to efficiently use it for the task
  - Bonus: Useful for several tasks and not just a specific task
Representing Words

- A 1-hot representation
  - tiger = \[0,0,0,0,0,0,0,1,0,0,0\]
  - lion = \[0,0,0,0,0,0,1,0,0,0,0\]
  - Vector with a single non-zero dimension
  - Representation does not capture similarity between words!
Distributional Method

A word is known by the company it keeps – John Rupert Firth

certainly open and the moon shining in on the
and the cold, close moon". And neither on
the night with the moon shining so bright
in the light of the moon. It all boils down
ly under a crescent moon, thrilled by ice
the seasons of the moon? Home, alone,
dazzling snow, the moon has risen full and
in the temple of the moon, driving out of

<table>
<thead>
<tr>
<th></th>
<th>planet</th>
<th>night</th>
<th>full</th>
<th>shadow</th>
<th>shine</th>
<th>crescent</th>
</tr>
</thead>
<tbody>
<tr>
<td>moon</td>
<td>10</td>
<td>22</td>
<td>43</td>
<td>16</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>sun</td>
<td>14</td>
<td>10</td>
<td>4</td>
<td>15</td>
<td>45</td>
<td>0</td>
</tr>
<tr>
<td>dog</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Representing Words - Brown Clusters

- Hierarchical clustering of words (based on classes)
- Discrete representation
- Very competitive and popular
- Useful for variety of tasks like NER, POS tagging etc

[Image from: https://www.researchgate.net/figure/261610872_fig1_A-hierarchical-structure-fragment-generated-by-Brown-clustering-for-7-words-from-the]
Distributional Method-Fundamentals

A word is known by the company it keeps – John Rupert Firth

1. I like Deep Learning
2. I enjoy flying
3. I like NLP

<table>
<thead>
<tr>
<th>counts</th>
<th>I</th>
<th>like</th>
<th>enjoy</th>
<th>deep</th>
<th>learning</th>
<th>NLP</th>
<th>flying</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>like</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>enjoy</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>deep</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>learning</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>NLP</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>flying</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>.</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

[Courtesy:Socher]
Distributional Method- Problems with Raw Co-occurrence Matrices

- Very high dimensional. Increases with vocabulary size.
- Less robust models due to data sparsity.
- Store important information in a fixed dimension dense vector.

[Courtesy: Socher]
Distributed Word Representations

Word Embeddings are latent representations of words.

We factorize the co-occurrence matrix.

Can be viewed as an online implicit factorizing method and thus scalable.

Implicit Dimensions: Similar words share similar representations.

Explicit Dimensions:

|V|: size of vocabulary

Latent Dimensions:

d << |V|

Similar words share similar representations.
SVD Word Embeddings

<table>
<thead>
<tr>
<th>counts</th>
<th>I</th>
<th>like</th>
<th>enjoy</th>
<th>deep</th>
<th>learning</th>
<th>NLP</th>
<th>flying</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>like</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>enjoy</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>deep</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>learning</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>NLP</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>flying</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>.</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

SVD

[U] [Σ] [V]

[Courtesy: Socher]
SVD Word Embeddings (Visualization)

Corpus: I like deep learning. I like NLP. I enjoy flying.
Printing first two columns of $U$ corresponding to the 2 biggest singular values

```python
for i in xrange(len(words)):
    plt.text(U[i,0], U[i,1], words[i])
```
Issues with Word Embeddings

- Computational Scalability: $O(n^3)$
- Does not scale well when we have millions of words.
- Might need to apply transformations on raw co-occurrence matrices (PPMI etc) to obtain high quality embeddings
An alternative approach: Neural Word Embeddings

- Learn word embeddings directly from data
- Use a neural network based architecture
- Online, scalable to large data sets
- Implicitly factorizes the co-occurrence matrix
Skipgram model – Learning Word Embeddings

V: Vocabulary, k: Embedding size

Learn parameters W (embeddings) and X.

Given a word w and a context word c, maximize Pr(c|w).

https://ronxin.github.io/wevi/
Visualizing word embeddings

Learn a mapping from words to a continuous space.
Visualizing Word Embeddings - Word Network
Interesting clusters
Summary

- Word Embeddings are learned directly from data
- Represent words in a low dimensional space capturing similarity in meaning
- Shown to be useful features for several NLP Tasks
- Scale well to large data
THANK YOU