Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
Background: Neural Languages Models

\[ Pr[w|context] = Pr[w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n+1}] \]
Background: Neural Language Models

\[ P_r[w|\text{context}] = P_r[w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n+1}] \]

- Each \( w_t \) parameterized with a set of values in a vector.
Background: Neural Language Models

\[ Pr[w|context] = Pr[w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n+1}] \]

- Each \( w_t \) parameterized with a set of values in a vector
- Bengio, 2003 - neural language model:
- Learning word representations stored lookup table/matrix
Background: Neural Language Models

\[ Pr[w|\text{context}] = Pr[w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n+1}] \]

- Each \( w_t \) parameterized with a set of values in a vector
- Bengio, 2003 - neural language model:
- Learning word representations stored lookup table/matrix
Background: Neural Language Models

\[ Pr[w|context] = Pr[w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n+1}] \]

- Impractical--extremely expensive fully-connected softmax layer
- Learned embeddings not transferable to other tasks
Background: Collobert & Weston (2008, 2011)

- Improved the objective function and removed expensive softmax layer
Background: Collobert & Weston (2008/2011)

- Improved the objective function and removed expensive softmax layer
- CNNs + tagging → semantic embeddings
Background: Collobert & Weston (2008/2011)

- Improved the objective function and removed expensive softmax layer
- CNNs + tagging → semantic embeddings
- Showed that learned embeddings could be useful for downstream tasks
Popular Embedding Tools

- Word2vec (Mikolov, 2013)
- GloVe (Pennington, 2014)
- fastTest (Facebook, 2015)
Popular Embedding Tools

- Word2vec (Mikolov, 2013)
- GloVe (Pennington, 2014)
- fastTest (Facebook, 2015)

Common Issues:

- Slow fully-connected layers
- Limited to text sequences
- Can we embed, say, documents and labels in a common vector space?
Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
Motivation of StarSpace

- Improve upon word2vec, fastText, and GloVe
Motivation of StarSpace

- Improve upon word2vec, fastText, and GloVe
- Generalizable ML: “Embed all the things”--not just text
  - Documents, words, sentences, labels, users, items to recommend to users, images
Motivation of StarSpace

- Improve upon word2vec, fastText, and GloVe
- Generalizable ML: “Embed all the things”--not just text
  - Documents, words, sentences, labels, users, items to recommend to users, images
- Embed entities of “Type A” with related entities of “Type B”
Motivation of StarSpace

- Improve upon word2vec, fastText, and GloVe
- Generalizable ML: “Embed all the things”--not just text
  - Documents, words, sentences, labels, users, items to recommend to users, images
- Embed entities of “Type A” with related entities of “Type B”
- Provide good (not necessarily best) performance for many tasks
Motivation of StarSpace

- Improve upon word2vec, fastText, and GloVe
- Generalizable ML: “Embed all the things”—not just text
  - Documents, words, sentences, labels, users, items to recommend to users, images
- Embed entities of “Type A” with related entities of “Type B”
- Provide good (not necessarily best) performance for many tasks
- StarSpace can be a goto baseline; tool you can try out on lots of problems
Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
Entities in StarSpace

- Entity: words, sentences, documents, users, images, labels, etc.
Entities in StarSpace

- Entity: words, sentences, documents, users, images, labels, etc.
- Old way: words represented as a single word ID
- Raw sentence: [huge iceberg in Greenland] ➔ [60, 100, 4, 55]
Entities in StarSpace

- Entity: words, sentences, documents, users, images, labels, etc.
- Old way: words represented as a single word ID
- Raw sentence: [huge iceberg in Greenland] → [60, 100, 4, 55]
- Embedding($w_i$) = LookupTable[i] = $[\Theta_{i1}, \Theta_{i2}, ..., \Theta_{i300}]^T$
Entities in StarSpace

- Entity: words, sentences, documents, users, images, labels, etc.
- Old way: words represented as a single word ID
- Raw sentence: [huge iceberg in Greenland] ➔ [60, 100, 4, 55]
- Embedding($w_i$) = LookupTable[$i$] = $[\Theta_{i1}, \Theta_{i2}, \ldots, \Theta_{i300}]^T$
- StarSpace: entities are bags-of-features (sets of feature ID’s)
Entities in StarSpace

- Entity: words, sentences, documents, users, images, labels, etc.
- Old way: words represented as a single word ID
- Raw sentence: [huge iceberg in Greenland] → [60, 100, 4, 55]
- \( \text{Embedding}(w_i) = \text{LookupTable}[i] = [\Theta_{i1}, \Theta_{i2}, ..., \Theta_{i300}]^T \)
- StarSpace: *entities* are bags-of-features (sets of feature ID’s)
- Entity \( a = [60, 100, 4, 55] \)
- \( \text{Embedding}(a) = \text{LookupTable}[60] + ... + \text{LookupTable}[55] \)
Entities in StarSpace

Dictionary F
Dimensions: D x d

D
Entities in StarSpace

Dictionary $F$

Dimensions: $D \times d$

$F[i]$
Entities in StarSpace

\[ F[i] \rightarrow \text{Dictionary } F \]
Dimensions: Dxd
Entities in StarSpace

Dictionary $F$
Dimensions: $D \times d$

$F[i] \rightarrow D$
Entities in StarSpace

Dictionary $F$ Dimensions: $D \times d$

$F[i]$ →

$\text{Embedding}(a) = \sum_{i \in a} F[i]$
Entities in StarSpace

- Embed “Type A” entities and “Type B” entities in the same vector space
  - (a, b) = (document, label)
  - (a, b) = (user, item to recommend)
  - (a, b) = (sentence, sentence)
Entities in StarSpace

- Embed “Type A” entities and “Type B” entities in the same vector space
  - (a, b) = (document, label)
  - (a, b) = (user, item to recommend)
  - (a, b) = (sentence, sentence)
- Document: bag of words
- Label: singleton feature (a word)
Entities in StarSpace

- Embed “Type A” entities and “Type B” entities in the same vector space
  - \((a, b) = (\text{document, label})\)
  - \((a, b) = (\text{user, item to recommend})\)
  - \((a, b) = (\text{sentence, sentence})\)
- Document: bag of words
- Label: singleton feature (a word)
- User: bag of items they’ve liked
- Item to recommend: single feature (e.g., a Facebook page)
Loss Function

\[ \sum_{(a,b) \in E^+, \ b^- \in E^-} L^{batch}(\text{sim}(a, b), \text{sim}(a, b^-_1), \ldots, \text{sim}(a, b^-_k)) \]
Loss Function

\[ \sum_{(a,b) \in E^+, \ b^- \in E^-} \text{\text{\text{}batch}}(\text{sim}(a, b), \text{sim}(a, b^-_1), \ldots, \text{sim}(a, b^-_k)) \]
Loss Function

\[
\sum_{(a,b) \in E^+, \ b^- \in E^-} L_{batch}^{(sim(a, b), \ sim(a, b^-_1), \ldots, \ sim(a, b^-_k))}
\]
Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
## Results

- **StarSpace vs fastText on Wikipedia dataset**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Hits@1</th>
<th>Hits@10</th>
<th>Hits@20</th>
<th>Mean Rank</th>
<th>Training Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Unsupervised methods</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TFIDF</td>
<td>24.79%</td>
<td>35.53%</td>
<td>38.25%</td>
<td>2523.68</td>
<td>-</td>
</tr>
<tr>
<td>fastText (public Wikipedia model)</td>
<td>5.77%</td>
<td>14.08%</td>
<td>17.79%</td>
<td>2393.38</td>
<td>-</td>
</tr>
<tr>
<td>fastText (our dataset)</td>
<td>5.47%</td>
<td>13.54%</td>
<td>17.60%</td>
<td>2363.74</td>
<td>40h</td>
</tr>
<tr>
<td>StarSpace (word-level training)</td>
<td>5.89%</td>
<td>16.41%</td>
<td>20.60%</td>
<td>1614.21</td>
<td>45h</td>
</tr>
<tr>
<td><strong>Supervised methods</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM Ranker BoW features</td>
<td>26.36%</td>
<td>36.48%</td>
<td>39.25%</td>
<td>2368.37</td>
<td>-</td>
</tr>
<tr>
<td>SVM Ranker: fastText features (public)</td>
<td>5.81%</td>
<td>12.14%</td>
<td>15.20%</td>
<td>1442.05</td>
<td>-</td>
</tr>
<tr>
<td>StarSpace (sentence pair training)</td>
<td>30.07%</td>
<td>50.89%</td>
<td>57.60%</td>
<td>422.00</td>
<td>36h</td>
</tr>
<tr>
<td>StarSpace (word+sentence training)</td>
<td>25.54%</td>
<td>45.21%</td>
<td>52.08%</td>
<td>484.27</td>
<td>69h</td>
</tr>
</tbody>
</table>
Results (10 tasks)

- StarSpace word and sentence-level models individually underperformed compared to word2vec and GloVe
  - Word2vec or GloVe had higher accuracy for 8/10 tests
Results (10 tasks)

- StarSpace word and sentence-level models individually underperformed compared to word2vec and GloVe
  - Word2vec or GloVe had higher accuracy for 8/10 tests
- Word + sentence models did better
Results (10 tasks)

- StarSpace word and sentence-level models individually underperformed compared to word2vec and GloVe
  - Word2vec or GloVe had higher accuracy for 8/10 tests
- Word + sentence models did better
- Ensemble word + sentence often even better
  - Best accuracy for 4/10 tests
Outline

1. Background: Language Models and Previous Embedding Algos
2. Motivation of StarSpace
3. What is StarSpace? What is new about it?
4. Results
5. Conclusions
Conclusions

- StarSpace allows greater generality and flexibility
- Succeeds at providing a reasonable baseline for many problems
- Not very efficient—doesn’t use hierarchical classification
- Discrete features, not continuous features