A La Carte Embedding: Cheap but Effective Induction of Semantic Feature Vectors

Mikhail Khodak*,1, Nikunj Saunshi*,1, Yingyu Liang2, Tengyu Ma3, Brandon Stewart1, Sanjeev Arora1

1: Princeton University, 2: University of Wisconsin-Madison, 3: FAIR/Stanford University
Motivations

Distributed representations for words / text have had lots of successes in NLP (language models, machine translation, text classification)
Motivations

Distributed representations for words / text have had lots of successes in NLP (language models, machine translation, text classification)

Motivations for our work:

• Can we induce embeddings for all kinds of features, especially those with very few occurrences (e.g. ngrams, rare words)
Motivations

Distributed representations for words / text have had lots of successes in NLP (language models, machine translation, text classification)

Motivations for our work:

• Can we induce embeddings for all kinds of features, especially those with very few occurrences (e.g. ngrams, rare words)

• Can we develop simple methods for unsupervised text embedding that compete well with state-of-the-art LSTM methods
Motivations

Distributed representations for words and text have had lots of successes in NLP (language models, machine translation, text classification) -

Motivations for our work:
- Can we induce embeddings for all kinds of features, especially those with very few occurrences (e.g. ngrams, rare words)
- Can we develop simple methods for unsupervised text embedding that compete well with state-of-the-art LSTM methods

We make progress on both problems:
- Simple and efficient method for embedding features (ngrams, rare words, synsets)
- Simple text embeddings using ngram embeddings which perform well on classification tasks
Word embeddings

• Core idea: Cooccurring words are trained to have high inner product
  • E.g. LSA, word2vec, GloVe and variants
Word embeddings

• Core idea: Cooccurring words are trained to have high inner product
  • E.g. LSA, word2vec, GloVe and variants

• Require few passes over a very large text corpus and do non-convex optimization

\[ \nu_w \in \mathbb{R}^d \]

word embeddings
Word embeddings

• Core idea: Cooccurring words are trained to have high inner product
  • E.g. LSA, word2vec, GloVe and variants

• Require few passes over a very large text corpus and do non-convex optimization

• Used for solving analogies, language models, machine translation, text classification …
Feature embeddings

• Capturing meaning of other natural language features
  • E.g. ngrams, phrases, sentences, annotated words, synsets
Feature embeddings

• Capturing meaning of other natural language features
  • E.g. ngrams, phrases, sentences, annotated words, synsets

• Interesting setting: features with zero or few occurrences
Feature embeddings

• Capturing meaning of other natural language features
  • E.g. ngrams, phrases, sentences, annotated words, synsets

• Interesting setting: features with zero or few occurrences

• One approach (extension of word embeddings): Learn embeddings for all features in a text corpus
Feature embeddings

Issues

• Usually need to learn embeddings for all features together
  • Need to learn many parameters
  • Computation cost paid is *prix fixe* rather than *à la carte*

• Bad quality for *rare features*
Feature embeddings

Firth revisited: Feature derives meaning from \textit{words} around it
Feature embeddings

Firth revisited: Feature derives meaning from words around it

Given a feature $f$ and one (few) context(s) of words around it, can we find a reliable embedding for $f$ efficiently?
Feature embeddings

Firth revisited: Feature derives meaning from **words** around it

Given a feature \( f \) and one (few) context(s) of words around it, can we find a reliable embedding for \( f \) efficiently?

Scientists attending ACL work on **cutting edge** research in NLP

**Petrichor**: the earthy scent produce when rain falls on dry soil

Roger Federer won the first **set** of the match
Problem setup

Given: Text corpus and high quality word embeddings trained on it

\[ v_w \in \mathbb{R}^d \]

Input: A feature in context(s)
Output: Good quality embedding for the feature

\[
\begin{align*}
    w_1 & \ldots \ w_j & f & w_{j+1} & \ldots & w_n \\
    \downarrow & & & & & \downarrow \\
    \text{Algorithm} & & & & & \Rightarrow \\
    & & & & & v_f \in \mathbb{R}^d
\end{align*}
\]
Linear approach

- Given a feature f and words in a context c around it

\[ v_f^{avg} = \frac{1}{|c|} \sum_{w \in c} v_w \]
Linear approach

• Given a feature f and words in a context c around it

\[ v_f^{avg} = \frac{1}{|c|} \sum_{w\in c} v_w \]

• Issues
  • stop words (“is”, “the”) are frequent but are less informative
  • Word vectors tend to share common components which will be amplified
Potential fixes

• Ignore stop words
Potential fixes

• Ignore stop words

• SIF weights\(^1\): Down-weight frequent words (similar to tf-idf)

\[
v_f = \frac{1}{|c|} \sum_{w \in c} \alpha_w \, v_w
\]

\[
\alpha_w = \frac{a}{a + p_w}
\]

\(p_w\) is frequency of \(w\) in corpus

---

1: Arora et al. ‘17
Potential fixes

• Ignore stop words

• SIF weights\(^1\): Down-weight frequent words (similar to tf-idf)

\[
v_f = \frac{1}{|c|} \sum_{w \in c} \alpha_w v_w
\]

\[
\alpha_w = \frac{a}{a + p_w}
\]

\(p_w\) is frequency of \(w\) in corpus

• All-but-the-top\(^2\): Remove the component of top direction from word vectors

\[
v_f = \frac{1}{|c|} \sum_{w \in c} v'_w = (I - uu^T)v_w^{avg}
\]

\[
u = top\_direction(\{v_w\})
\]

\[
v'_w = remove\_component(v_w, u)
\]

1: Arora et al. ‘17, 2: Mu et al. ‘18
Our more general approach

• Down-weighting and removing directions can be achieved by matrix multiplication

\[ v_f \approx A \frac{1}{|c|} \sum_{w \in c} v_w = Av_f^{avg} \]
Our more general approach

- Down-weighting and removing directions can be achieved by matrix multiplication

\[ v_f \approx A \frac{1}{|c|} \sum_{w \in c} v_w = \text{Induced Embedding} \]

- Learn \( A \) by using words as features

\[ A^* = \text{argmin}_A \sum_{w} |v_w - A v_w^{\text{avg}}|^2 \]

- Learn \( A \) by \textbf{linear regression} and is unsupervised
Theoretical justification

• [Arora et al. TACL ’18] prove that under a generative model for text, there exists a matrix $A$ which satisfies

$$v_w \approx A v_w^{avg}$$
Theoretical justification

• [Arora et al. TACL ’18] prove that under a generative model for text, there exists a matrix $A$ which satisfies

$$v_w \approx A v_{w}^{avg}$$

• Empirically we find that the best $A^*$ recovers the original word vectors

$$\text{cosine}(v_w, A^* v_{w}^{avg}) \geq 0.9$$
A la carte embeddings

1. Learn induction matrix

\[ A^* = \arg\min_A \sum_w |v_w - A v_{avg}^w|^2 \]
A la carte embeddings

1. Learn induction matrix

\[ A^* = \arg \min_A \sum_w |v_w - A v_w^{avg}|^2 \]

2. A la carte embeddings

\[ v_f^{alc} = A^* v_f^{avg} = A^* \left( \frac{1}{|c|} \sum_{w \in c} v_w \right) \]
A la carte embeddings

1. Learn induction matrix

\[ A^* = \text{arg} \min_A \sum_w |v_w - A v_{w}^{\text{avg}}|^2 \]

2. A la carte embeddings

\[ v_f^{\text{alc}} = A^* v_f^{\text{avg}} = A^* \left( \frac{1}{|c|} \sum_{w \in c} v_w \right) \]
Advantages

• **à la carte:** Compute embedding only for given feature

• **Simple optimization:** Linear regression

• **Computational efficiency:** One pass over corpus and contexts

• **Sample efficiency:** Learn only $d^2$ parameters for $A^*$ (rather than $Vd$)

• **Versatility:** Works for any feature which has at least 1 context
Effect of induction matrix

• We plot the extent to which $A^*$ down-weights words against frequency of words compared to all-but-the-top
Effect of induction matrix

- We plot the extent to which $A^*$ down-weights words against frequency of words compared to all-but-the-top

$A^*$ mainly down-weights words with very high and very low frequency

All-but-the-top mainly down-weights frequent words
Effect of number of contexts

**Contextual Rare Words (CRW) dataset** providing contexts for rare words

- Task: Predict human-rated similarity scores for pairs of words
- Evaluation: Spearman’s rank coefficient between inner product and score

1: Subset of RW dataset [Luong et al. ’13]
Effect of number of contexts

**Contextual Rare Words (CRW) dataset**\(^1\) providing contexts for rare words
- Task: Predict human-rated similarity scores for pairs of words
- Evaluation: Spearman’s rank coefficient between inner product and score

Compare to the following methods:
- Average of words in context
- Average of non stop words
- SIF weighted average
- all-but-the-top

---

1: Subset of RW dataset [Luong et al. ’13]
Nonce definitional task

• Task: Find embedding for unseen word/concept given its definition
• Evaluation: Rank of word/concept based on cosine similarity with true embedding

iodine: is a chemical element with symbol I and atomic number 53

1: Herbelot and Baroni ‘17
Nonce definitional task

• Task: Find embedding for unseen word/concept given its definition
• Evaluation: Rank of word/concept based on cosine similarity with true embedding

iodine: is a chemical element with symbol I and atomic number 53

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean Reciprocal Rank</th>
<th>Median Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>word2vec</td>
<td>0.00007</td>
<td>111012</td>
</tr>
<tr>
<td>average</td>
<td>0.00945</td>
<td>3381</td>
</tr>
<tr>
<td>average, no stop words</td>
<td>0.03686</td>
<td>861</td>
</tr>
<tr>
<td>nonce2vec¹</td>
<td>0.04907</td>
<td>623</td>
</tr>
<tr>
<td>à la carte</td>
<td>0.07058</td>
<td>165.5</td>
</tr>
</tbody>
</table>

1: Herbelot and Baroni ‘17

modified version of word2vec
Ngram embeddings

Induce embeddings for ngrams using contexts from a text corpus

We evaluate the quality of embedding for a bigram \( f = (w_1, w_2) \) by looking at closest words to this embedding by cosine similarity.

<table>
<thead>
<tr>
<th>Method</th>
<th>beef up</th>
<th>cutting edge</th>
<th>harry potter</th>
<th>tight lipped</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_f^{add} = v_{w_1} + v_{w_2} )</td>
<td>meat, out</td>
<td>cut, edges</td>
<td>deathly, azkaban</td>
<td>loose, fitting</td>
</tr>
<tr>
<td>( v_f^{avg} )</td>
<td>but, however</td>
<td>which, both</td>
<td>which, but</td>
<td>but, however</td>
</tr>
<tr>
<td>ECO(^1)</td>
<td>meats, meat</td>
<td>weft, edges</td>
<td>robards, keach</td>
<td>scaly, bristly</td>
</tr>
<tr>
<td>Sent2Vec(^2)</td>
<td>\textit{add}, reallocate</td>
<td>\textit{science}, multidisciplinary</td>
<td>naruto, pokemon</td>
<td>wintel, codebase</td>
</tr>
<tr>
<td>( \text{à la carte} (A^*v_f^{avg}) )</td>
<td>need, \textit{improve}</td>
<td>\textit{innovative}, technology</td>
<td>\textit{deathly}, hallows</td>
<td>\textit{worried}, very</td>
</tr>
</tbody>
</table>

1: Poliak '17, 2: Pagliardini et al. '18
Unsupervised text embeddings

This movie is great!

\[ \begin{aligned}
    v &\in \mathbb{R}^d \\
    \mathbf{v} &\in \mathbb{R}^d
\end{aligned} \]
Unsupervised text embeddings

This movie is great!

Sparse
Bag-of-words, Bag-of-ngrams
Good performance

LSTM
Predict surrounding words / sentences
SOTA on some tasks

Linear
Sum of word/ngram embeddings
Compete with Bag-of-ngrams and LSTMs on some tasks
A la carte text embeddings

Linear schemes are typically weighted sums of ngram embeddings
A la carte text embeddings

Linear schemes are typically weighted sums of ngram embeddings

Types of ngrams embeddings

- DisC
- ECO
- Sent2Vec
- A La Carte

Compositional
- Flexible

Learned
- High quality
A la carte text embeddings

Linear schemes are typically weighted sums of ngram embeddings

A La Carte text embeddings are as concatenations of sum of à la carte ngram embeddings (as in DisC)

$$v^n_{document} = \left[ \sum v_{word}, \sum v^\text{alc}_{bigrant}, \ldots, \sum v^\text{alc}_{ngram} \right]$$
# A la carte text embeddings

\[ \nu_{\text{document}}^{\text{alc}} = \left[ \sum \nu_{\text{word}}^{\text{alc}} , \sum \nu_{\text{bigram}}^{\text{alc}} , \ldots , \sum \nu_{\text{ngram}}^{\text{alc}} \right] \]

<table>
<thead>
<tr>
<th>Method</th>
<th>n</th>
<th>dimension</th>
<th>MR</th>
<th>CR</th>
<th>SUBJ</th>
<th>MPQA</th>
<th>TREC</th>
<th>SST (±1)</th>
<th>SST</th>
<th>IMDB</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sparse</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bag-of-ngrams</td>
<td>1-3</td>
<td>100K-1M</td>
<td>77.8</td>
<td>78.3</td>
<td>91.8</td>
<td>85.8</td>
<td>90.0</td>
<td>80.9</td>
<td>42.3</td>
<td>89.8</td>
</tr>
<tr>
<td>Skip-thoughts(^1)</td>
<td></td>
<td></td>
<td>80.3</td>
<td>83.8</td>
<td>94.2</td>
<td>88.9</td>
<td>93.0</td>
<td>85.1</td>
<td>45.8</td>
<td></td>
</tr>
<tr>
<td>SDAE(^2)</td>
<td>2400</td>
<td></td>
<td>74.6</td>
<td>78.0</td>
<td>90.8</td>
<td>86.9</td>
<td>78.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN-LSTM(^3)</td>
<td></td>
<td></td>
<td>77.8</td>
<td>82.0</td>
<td>93.6</td>
<td>89.4</td>
<td>92.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MC-QT(^4)</td>
<td>4800</td>
<td></td>
<td>82.4</td>
<td>86.0</td>
<td>94.8</td>
<td>90.2</td>
<td>92.4</td>
<td>87.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LSTM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DisC(^5)</td>
<td>2-3</td>
<td>\leq 4800</td>
<td>80.1</td>
<td>81.5</td>
<td>92.6</td>
<td>87.9</td>
<td>90.0</td>
<td>85.5</td>
<td>46.7</td>
<td>89.6</td>
</tr>
<tr>
<td>Sent2Vec(^6)</td>
<td>1-2</td>
<td>700</td>
<td>76.3</td>
<td>79.1</td>
<td>91.2</td>
<td>87.2</td>
<td>85.8</td>
<td>80.2</td>
<td>31.0</td>
<td>85.5</td>
</tr>
<tr>
<td>à la carte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2400</td>
<td><strong>81.3</strong></td>
<td>83.7</td>
<td>93.5</td>
<td>87.6</td>
<td>89.0</td>
<td><strong>85.8</strong></td>
<td>47.8</td>
<td>90.3</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>4800</td>
<td><strong>81.8</strong></td>
<td>84.3</td>
<td><strong>93.8</strong></td>
<td>87.6</td>
<td>89.0</td>
<td>86.7</td>
<td><strong>48.1</strong></td>
<td>90.9</td>
</tr>
</tbody>
</table>

1: Kiros et al. ‘15, 2: Hill et al. ‘16, 3: Gan et al. ‘17, 4: Logeswaran and Lee ‘18, 5: Arora et al. ‘18, 6: Pagliardini et al. ‘18
Conclusions

• Simple and efficient method for inducing embeddings for many kinds of features, given at least one context of usage

• Embeddings produced are in same semantic space as word embeddings

• Good empirical performance for rare words, ngrams and synsets

• Text embeddings that compete with unsupervised LSTMs

Code is on github: https://github.com/NLPrinceton/ALaCarte
CRW dataset available: http://nlp.cs.princeton.edu/CRW/
Future work

- Zero shot learning of feature embeddings
  - Compositional approaches

- Harder to annotate features (synsets)

- Contexts based on other syntactic structures
Thank you!

Questions?

{nsaunshi, mkhodak}@cs.princeton.edu