Deep Learning for Natural Language Processing

Inspecting and evaluating word embedding models

UNIVERSITY OF GOTHENBURG

CHALMERS

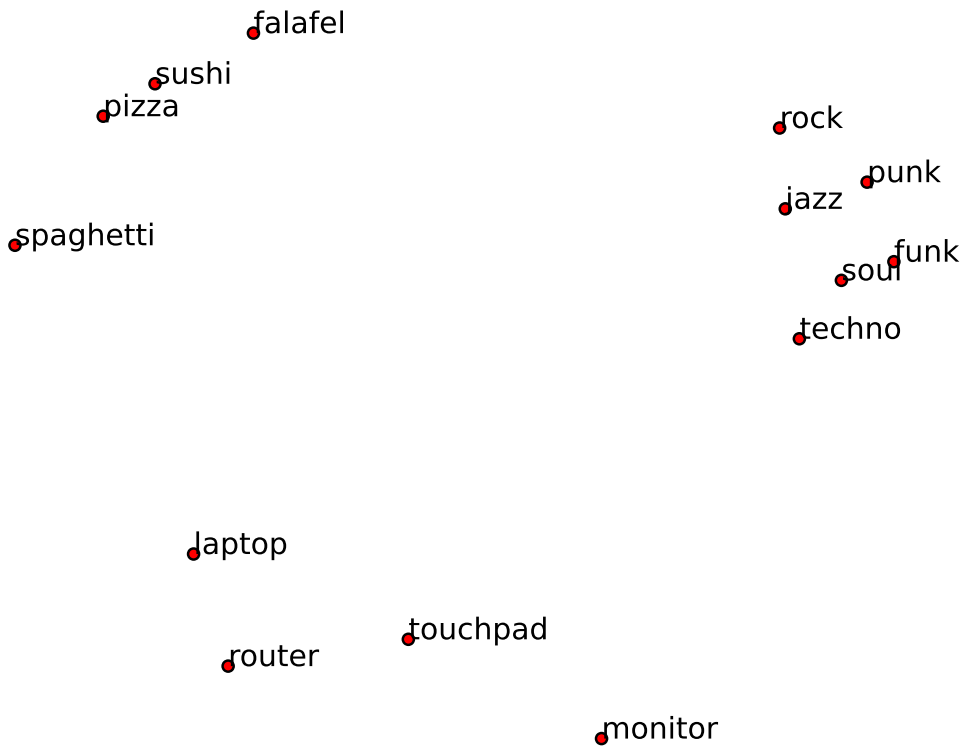
WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

Richard Johansson

richard.johansson@gu.se

inspection of the model

- after training the embedding model, we can inspect the result for a qualitative interpretation
- for illustration, vectors can be projected to two dimensions using methods such as t-SNE or PCA



computing similarities

- another method for inspecting embeddings is based on computing **similarities**
- most commonly, the **cosine similarity**:

$$\text{cos-sim}(x, y) = \frac{x \cdot y}{\sqrt{\|x\|^2 \cdot \|y\|^2}}$$

- this allows us to compare relative similarity scores:

```
word_vectors_pretrained.similarity('tomato', 'cucumber')
```

```
0.6310712
```

```
word_vectors_pretrained.similarity('tomato', 'computer')
```

```
0.09678186
```

```
word_vectors_pretrained.similarity('tomato', 'autonomous')
```

```
-0.03140637
```

nearest neighbor lists

- using a similarity or distance function, we can find a set of **nearest neighbors**:

```
10 most similar to 'tomato':

tomatoes          0.8442
lettuce           0.7070
asparagus         0.7051
peaches           0.6939
cherry_tomatoes   0.6898
strawberry        0.6889
strawberries      0.6833
bell_peppers      0.6814
potato            0.6784
cantaloupe        0.6780
```

how do we measure how “good” the word embeddings are?

evaluation of word embedding models: high-level ideas

- **intrinsic** evaluation: use some benchmark to evaluate the embeddings directly
  - similarity benchmarks
  - synonymy benchmarks
  - analogy benchmarks
  - ...
- **extrinsic** evaluation: see which vector space works best in an application where it is used

comparing to a similarity benchmark

- how well do the similarities computed by the model work?

```
10 most similar to 'tomato':

tomatoes          0.8442
lettuce           0.7070
asparagus         0.7051
peaches           0.6939
cherry_tomatoes 0.6898
...
```

- if we have a list of word pairs where **humans have graded the similarity**, we can measure how well the similarities correspond

the WS-353 benchmark

```
Word 1,Word 2,Human (mean)
love,sex,6.77
tiger,cat,7.35
tiger,tiger,10.00
book,paper,7.46
computer,keyboard,7.62
computer,internet,7.58
plane,car,5.77
train,car,6.31
telephone,communication,7.50
television,radio,6.77
media,radio,7.42
drug,abuse,6.85
bread,butter,6.19
...
```

Spearman's rank correlation

- if we sort the similarity benchmark, and sort the similarities computed from our vector space, we get two ranked lists
- **Spearman's rank correlation coefficient** compares how much the ranks differ between two ranked lists:

$$r = 1 - \frac{6 \cdot \sum d_i^2}{n \cdot (n^2 - 1)}$$

  where d_i is the rank difference for word i, and n the number of items in the list
- the maximal value is 1, when the lists are identical

a few similarity benchmarks

- the WS-353 dataset has been criticized because it does not distinguish between **similarity** and **relatedness**
  - ***screen*** is **similar** to ***monitor***
  - ***screen*** is **related** to ***resolution***
- there are several other similarity benchmarks

No.	Task Name	Word pairs	Reference
1	WS-353	353	Finkelstein et. al, 2002
2	WS-353-SIM	203	Agirre et. al, 2009
3	WS-353-REL	252	Agirre et. al, 2009
4	MC-30	30	Miller and Charles, 1930
5	RG-65	65	R and G, 1965
6	Rare-Word	2034	Luong et. al, 2013
7	MEN	3000	Bruni et. al, 2012
8	MTurk-287	287	Radinsky et. al, 2011
9	MTurk-771	771	Halawi and Dror, 2012
10	YP-130	130	Yang and Powers, 2006
11	SimLex-999	999	Hill et. al, 2014
12	Verb-144	144	Baker et. al, 2014

- see e.g. https://github.com/vecto-ai/word-benchmarks

synonymy and antonymy test sets

- example from (Sahlgren, 2006):

Word	Alternatives	Synonym
spot	sea	
	location	√
	latitude	
	climate	

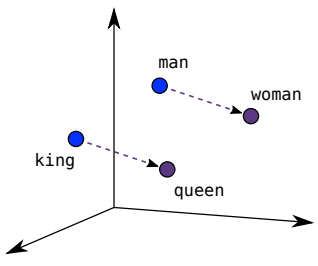
Table 12.1: TOEFL synonym test for "spot."

word analogies

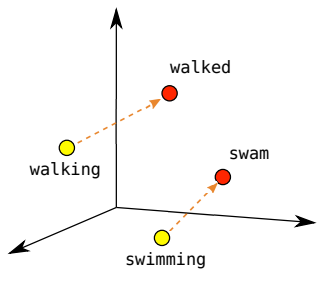
- **word analogy** (Google test set): *Moscow* is to *Russia* as *Copenhagen* is to $\boldsymbol{X}$?
  - in some vector space models, we can get a reasonably good answer by a simple vector operation:

$$V(\boldsymbol{X}) = V(Copenhagen) + (V(Russia) - V(Moscow))$$

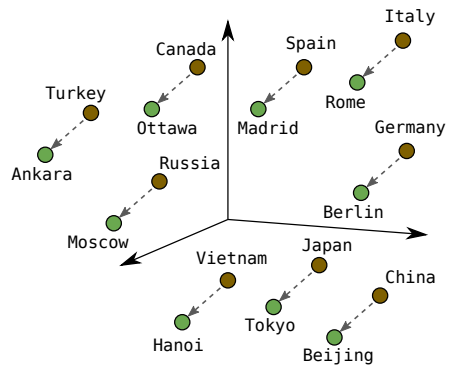
  - then find the word whose vector is closest to $V(\boldsymbol{X})$
  - see Mikolov et al. (2013)



Male-Female



Verb Tense



Country-Capital

[source]

extrinsic evaluation

- in **extrinsic evaluation**, we compare embedding models by "plugging" them into an application and comparing end results
  - categorizers, taggers, parsers, translation, ...
- **no reason to assume that one embedding model is always the "best"** (Schnabel et al., 2015)
  - depends on the application

do benchmarks for intrinsic evaluation predict application performance?

- short answer: not reliably
- Chiu et al. (2016) find that only one benchmark (SimLex999) correlates with tagger performance
- Faruqui et al. (2016) particularly criticizes the use of similarity benchmarks
- both papers are from the **RepEval** workshop
  - https://repeval2019.github.io/program/

references I

B. Chiu, A. Korhonen, and S. Pyysalo. 2016. Intrinsic evaluation of word vectors fails to predict extrinsic performance. In *RepEval*.

M. Faruqui, Y. Tsvetkov, P. Rastogi, and C. Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. In *RepEval*.

T. Mikolov, W.-t. Yih, and G. Zweig. 2013. Linguistic regularities in continuous space word representations. In *NAACL*.

M. Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm U.

T. Schnabel, I. Labutov, D. Mimno, and T. Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *EMNLP*.