Deep Learning for Natural Language Processing

More training methods for word embeddings

UNIVERSITY OF GOTHENBURG

CHALMERS

WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

Richard Johansson

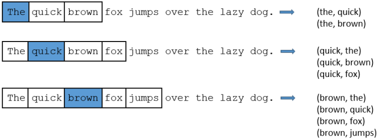
richard.johansson@gu.se

overview

- research on vector-based word representations goes back to the 1990s but took of in 2013 with the publication of the SGNS model
- while SGNS is probably the most well-known word embedding model, there are several others
- we'll take a quick tour of different approaches

training word embeddings: high-level approaches

- “**prediction-based**”: collecting training instances from individual occurrences (like SGNS)



- “**count-based**”: methods based on cooccurrence matrices

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

SGNS: recap

- in SGNS, our parameters are the **target word embeddings** V_T and the **context word embeddings** V_C
- **positive training examples** are generated by collecting word pairs, and **negative examples** by sampling contexts randomly
- we train the following model with respect to (V_T, V_C):

$$P(\text{true pair}|(w,c)) = \frac{1}{1 + \exp\left(-V_T(w) \cdot V_C(c)\right)}$$

$$P(\text{synthetic pair}|(w,c)) = 1 - \frac{1}{1 + \exp\left(-V_T(w) \cdot V_C(c)\right)}$$

continuous bag-of-words for training embeddings

- the continuous bag-of-words (CBoW) model considers the **whole context** instead of breaking it up into separate pairs:

  the quick brown ***fox*** *jumps over the lazy dog*

  $\Downarrow$

  { *the, quick, brown, jumps, over, the* }, ***fox***

- the model is almost like SGNS:

$$P(\text{true pair}|(w, C)) = \frac{1}{1 + \exp\left(-V_T(w) \cdot V_C(C)\right)}$$

  where $V_C(C)$ is the sum of context embeddings

$$V_C(C) = \sum_{c \in C} V_C(c)$$

- also available in the `word2vec` software

how can we deal with **out-of-vocabulary** words?

- what if *dingo* is in the vocabulary but not *dingoes*?
- humans can handle these kinds of situations!
- **fastText** (Bojanowski et al., 2017) modifies the SGNS model to handle these situations:

$$V_T(w) = \sum_{g \in \mathcal{G}} \boldsymbol{z}_g$$

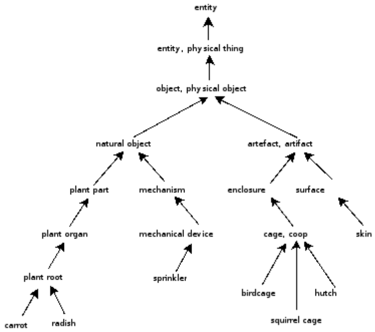
  where $\mathcal{G}$ is the set of subwords for w:

$$\mathcal{G} = \{\text{'<dingoes>', '<di', 'din', 'ing', ..., 'ngoes>' }\}$$

- handles rare words and OOV words better than SGNS

combining knowledge-based and data-driven representations

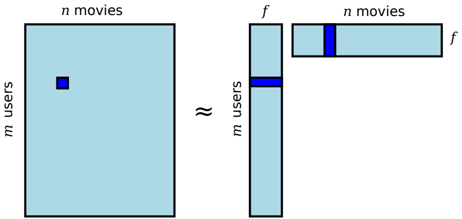
- in traditional AI ("GOFAI") and in linguistic theory, word meaning is expressed using some **knowledge representation**
- in NLP, **WordNet** is the most popular lexical knowledge base:



- Faruqui et al. (2015) "retrofits" word embeddings using a LKB
- Nieto Piña and Johansson (2017) propose a modified SGNS algorithm that uses a LKB to distinguish **senses**

perspective: matrix factorization in recommender systems

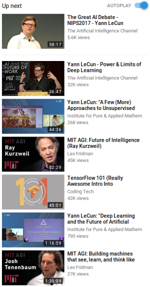
- the most famous approach in recommenders is based on **factorization** of the user/item rating matrix



- to predict a missing cell (rating of an unseen item):

$$\hat{r}_{ui} = \boldsymbol{p}_u \cdot \boldsymbol{q}_i$$

where $\boldsymbol{p}_u$ is the user's vector, and $\boldsymbol{q}_i$ the item's vector



example of a word–word co-occurrence matrix

- assume we have the following set of texts:
  - "I like NLP"
  - "I like deep learning"
  - "I enjoy flying"

$$
X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array}
\begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\
\begin{bmatrix}
0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{bmatrix} \end{array}
$$

[source]

matrix-based word embeddings

- **Latent Semantic Analysis** (Landauer and Dumais, 1997) was the first vector-based word representation model
  - it applies singular value decomposition (SVD) to a word–document matrix
- several variations of this approach:
  - counts stored in the matrix (word–document, word–word, ...)
  - transformations of the matrix (log, PMI, ...)
  - factorization of the matrix (none, SVD, NNMF, ...)

GloVe

- GloVe (Pennington et al., 2014) is a famous matrix-based word embedding training method
  - https://nlp.stanford.edu/projects/glove/
- they claim that their model trains more robustly than SGNS and they report better results on some benchmarks
- in GloVe, we try to find embeddings to reconstruct the log-transformed cooccurrence count matrix:

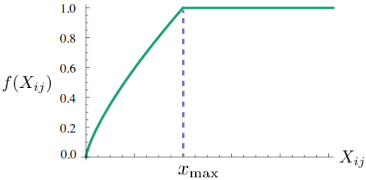
$$V_T(w) \cdot V_C(c) \approx \log X(w, c)$$

objective function in GloVe

- GloVe minimizes the following loss function over the cooccurrence matrix:

$$J = \sum_{w,c} f(X(w,c))\,(V_T(w) \cdot V_C(c) - \log X(w,c))^2$$

- the function f is used to downweight low-frequency words:



what should we prefer, count-based or prediction-based?

- see Baroni et al. (2014) for a comparison of count-based and prediction-based
  - they come out strongly in favor of prediction-based
  - but this result has been questioned
- pros and cons:
  - prediction-based methods are sensitive to the order the examples are processed
  - count-based methods can be messy to implement with a large vocabulary
- Levy and Goldberg (2014) show a connection between SGNS and matrix-based methods and the GloVe paper (Pennington et al., 2014) also discusses the connections

references

M. Baroni, G. Dinu, and G. Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. 2017. Enriching word vectors with subword information. *TACL* 5:135–146.

M. Faruqui, Y. Tsvetkov, D. Yogatama, C. Dyer, and N. A. Smith. 2015. Sparse overcomplete word vector representations. In *ACL*.

T. K. Landauer and S. T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review* 104:211–240.

O. Levy and Y. Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *NIPS*.

L. Nieto Piña and R. Johansson. 2017. Training word sense embeddings with lexicon-based regularization. In *IJCNLP*.

J. Pennington, R. Socher, and C. Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*.