INF5820
Distributional Semantics: Extracting Meaning from Data

# Lecture 4

# Beyond words: distributional representations of texts

Andrey Kutuzov
andreku@ifi.uio.no

16 November 2016



## Contents





## Brief recap

**What we are going to cover today**

- Problem of document classification;
- Traditional bag-of-words approach;
- Compositional distributed approaches;
- 'Proper' distributional approaches (document vectors).



## Contents

## Problem description

- Distributional approaches allow to extract semantics from unlabeled data at word level.
- But we also need to represent variable-length documents!
  - for classification,
  - for clustering,
  - for information retrieval (including web search).



## Problem description

- Can we detect semantically similar texts in the same way as we detect similar words?
- Yes we can!
- Nothing prevents us from representing sentences, paragraphs or whole documents (further we use the term '*document*' for all these things) as dense vectors.
- After the documents are represented as vectors, classification, clustering or other data processing tasks become trivial.
- Unsupervised learning of distributed representations for documents is the topic of this lecture.

Note: this lecture does not cover sequence-to-sequence sentence modeling approaches based on RNNs (LSTM, GRU, etc). A good example of those is the Skip-Thought algorithm [Kiros et al., 2015]. We are concerned with comparatively simple algorithms conceptually similar to prediction-based distributional models for words.



## Contents





## Can we do without semantics?

**Bag-of-words with TF-IDF**

A very strong baseline approach for document representation, hard to beat by modern methods:

1. Extract vocabulary $V$ of all words (terms) in the training collection consisting of $N$ documents;
2. For each term, calculate its document frequency: in how many documents it occurs ($df$);
3. Represent each document as a sparse vector of frequencies for all terms from $V$ contained in it ($tf$);
4. For each value, calculate the weighted frequency $wf$ using term frequency / inverted document frequency (TF-IDF):
   - $wf = (1 + log_{10} tf) \times log_{10}(\frac{N}{df})$
5. Use these weighted document vectors in your downstream tasks.

## Can we do without semantics?

### Bag-of-words problems

Unfortunately, simple bag-of-word does not take into account semantic relationships between linguistic entities.
No way to detect semantic similarity between documents which do not share words:

- *California saw mass protests after the elections.*
- *Many Americans were anxious about the elected president.*

It means we need more sophisticated semantically-aware distributed methods, like neural embeddings.



## Contents





## Distributed models: composing from word vectors

- Document meaning is composed of individual word meanings.
- Need to combine continuous word vectors into continuous document vectors.
- It is called a composition function.

### Semantic fingerprints

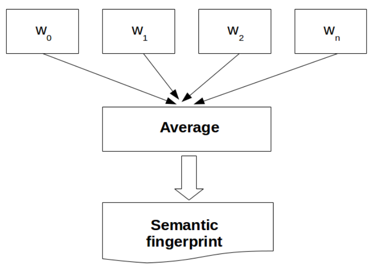
- One of the simplest composition functions: an average vector $\vec{S}$ over vectors of all words $w_0 ... n$ in the document.

$$\vec{S} = \frac{1}{n} \times \sum_{i=0}^{n} \vec{w_n} \tag{1}$$

- We don't care about syntax and word order.
- If we already have a good word embedding model, this bottom-up approach is strikingly efficient and usually beats bag-of-words.
- Let's call it a 'semantic fingerprint' of the document.
- It is very important to remove stop words beforehand!



## Distributed models: composing from word vectors




- You even don't have to average. Summing vectors is enough: cosine is about angles, not magnitudes.
- However, averaging makes difference in case of other distance metrics (Euclidean distance, etc).
- Also helps to keep things tidy and normalized.

### Composition functions

- One can experiment with different combinations of word vectors, not only averaging:
  - Concatenation
  - Multiplication
  - Weighted sum
  - etc...
- Can introduce word order knowledge by using n-grams instead of words.
- See [Mitchell and Lapata, 2010] for extensive description and evaluation of various compositional models.





### Advantages of semantic fingerprints

- *Semantic fingerprints* work fast and reuse already trained models.
- Generalized document representations do not depend on particular words.
- They take advantage of 'semantic features' learned during the model training.
- Topically connected words collectively increase or decrease expression of the corresponding semantic components.
- Thus, topical words automatically become more important than noise words.

See more in [Kutuzov et al., 2016].





### Deep inverse regression

- Another approach using the existing word vectors is proposed in [Taddy, 2015]:
- Classify documents by inverting distributional models:
  - Prediction-based models contain information on typical neighbors for all words in their output weight matrix;
  - Thus, we know how likely it is for two words to occur together;
  - It is then possible to detect the likelihood of a sentence given a model;
  - But it means we can employ Bayes rule to calculate the inverse of this: the likelihood of the model given a sentence;
- For example, we have models trained on positive and negative reviews;
- For any sentence we can calculate the probability of it being positive or negative.
- Deep inverse regression is implemented in *Gensim* (for hierarchical softmax models only);
- drawback: you need separate models for each of your classes.





### But...

However, for some problems such compositional approaches are not enough and we need to generate real document embeddings.

## Contents





## Distributed models: training document vectors

### Many questions

- Should we treat the document as a single target unit surrounded by its counterparts?
  - Usually not feasible: sentences are rarely repeated, let alone documents.
- Or should we take into account the composing words, but go beyond simple composition functions?



## Distributed models: training document vectors

### Paragraph Vector

- [Le and Mikolov, 2014] proposed Paragraph Vector;
- primarily designed for learning sentence vectors;
- the algorithm takes as an input sentences/documents tagged with (possibly unique) identifiers;
- learns distributed representations for the sentences, such that similar sentences have similar vectors;
- so each sentence is represented with an identifier and a vector, like a word;
- these vectors serve as sort of document memories or document topics.

Not much evaluated (however, see [Hill et al., 2016] and [Lau and Baldwin, 2016])



## Distributed models: training document vectors

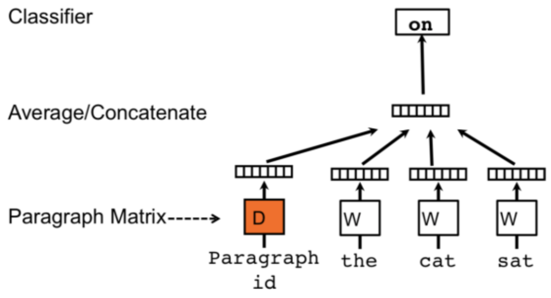
### Paragraph Vector (aka doc2vec)

- implemented in *Gensim* under the name *doc2vec*;
- Distributed memory (DM) and Distributed Bag-of-words (DBOW) methods;
- PV-DM:
  - learn word embeddings in a usual way (shared by all documents);
  - randomly initialize document vectors;
  - use document vectors together with word vectors to predict the neighboring words within a pre-defined window;
  - minimize error;
  - the trained model can inference a vector for any new document (the model remains intact).
- PV-DBOW:
  - don't use sliding window at all;
  - just predict all words in the current document using its vector.

Contradicting reports on which method is better.

# Distributed models: training document vectors

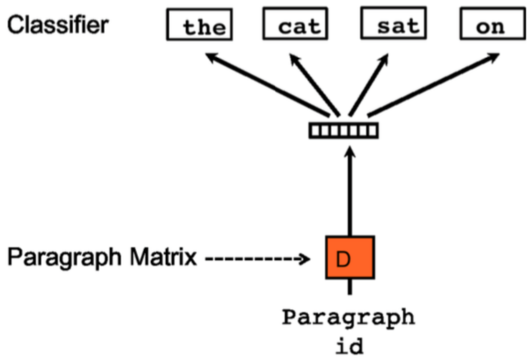
Paragraph Vector - Distributed memory (PV-DM)






# Distributed models: training document vectors

Paragraph Vector - Distributed Bag-of-words (PV-DBOW)






# Distributed models: training document vectors

**Paragraph Vector (aka doc2vec)**

- You train the model, then inference embeddings for the documents you are interested in.
- The resulting embeddings are shown to perform very good on sentiment analysis and other document classification tasks, as well as in IR tasks.
- Very memory-hungry: each sentence gets its own vector (many millions of sentences in the real-life corpora).
- It is possible to reduce the memory footprint by training a limited number of vectors: group sentences into classes.



# Distributed models: training document vectors

There can be many other ways to employ embeddings in document representation tasks!

For example:

- interpret sentences or paragraphs as 'words' and train a straightforward word embedding model.
- etc...

The choice of an approach depends very much on your downstream task.

## Distributed models: training document vectors

Questions?

INF5820
Distributional Semantics: Extracting Meaning from Data
Lecture 4
Beyond words: distributional representations of texts

Homework: obligatory assignment 3.



## Contents





## In the next week

Kings and queens, men and women: semantic relations between word embeddings

- Distributional models contain not only words, but also relations between them;
- Why is that so?
- Mathematics behind this;
- Possible applications of semantic relations in distributional models;
- Projecting one model into another;
- etc...



## References I

Hill, F., Cho, K., and Korhonen, A. (2016).
Learning distributed representations of sentences from unlabelled data.
*arXiv preprint arXiv:1602.03483*.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. (2015).
Skip-thought vectors.
In *Advances in neural information processing systems*, pages 3294–3302.

Kutuzov, A., Kopotev, M., Ivanova, L., and Sviridenko, T. (2016).
Clustering comparable corpora of Russian and Ukrainian academic texts: Word embeddings and semantic fingerprints.
In *Proceedings of the Ninth Workshop on Building and Using Comparable Corpora, held at LREC-2016*, pages 3–10. European Language Resources Association.

## References II


Lau, H. J. and Baldwin, T. (2016).
An empirical evaluation of doc2vec with practical insights into document embedding generation.
In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 78–86. Association for Computational Linguistics.

Le, Q. V. and Mikolov, T. (2014).
Distributed representations of sentences and documents.
In *ICML*, volume 14, pages 1188–1196.

Mitchell, J. and Lapata, M. (2010).
Composition in distributional models of semantics.
*Cognitive science*, 34(8):1388–1429.




## References III


Taddy, M. (2015).
Document classification by inversion of distributed language representations.
In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 45–49, Beijing, China.