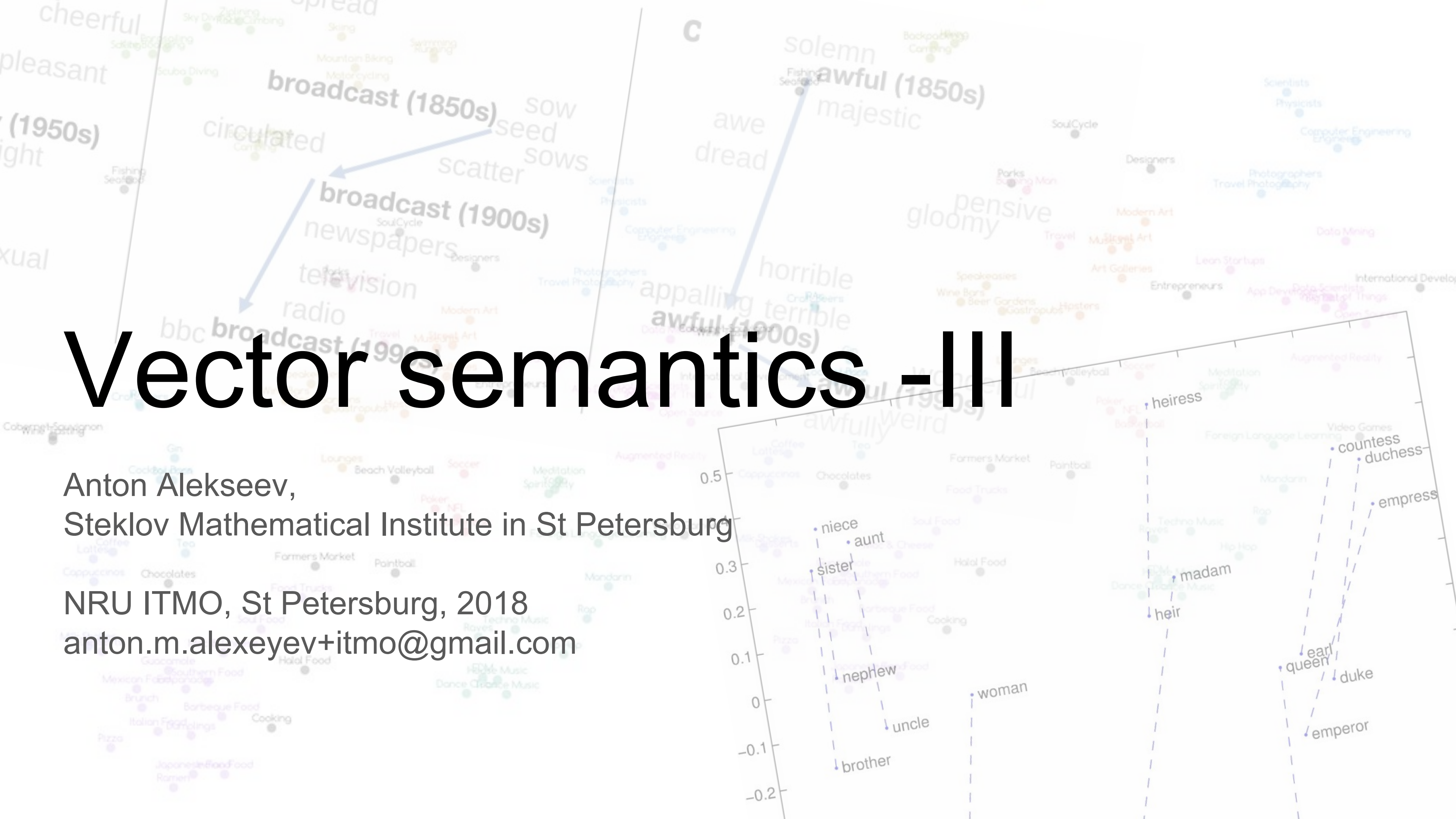


Vector semantics -III

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg

NRU ITMO, St Petersburg, 2018
anton.m.alexeyev+itmo@gmail.com

REMINDER

Distributional hypothesis

- Zellig S. Harris: “oculist and eye-doctor... occur in almost the same environments”, “If A and B have almost **identical environments**. . . we say that they are synonyms”

- Most famous, John Firth: **You shall know a word by the company it keeps!**

BTW, Z. Harris is sometimes referred to as Noam Chomsky’s teacher

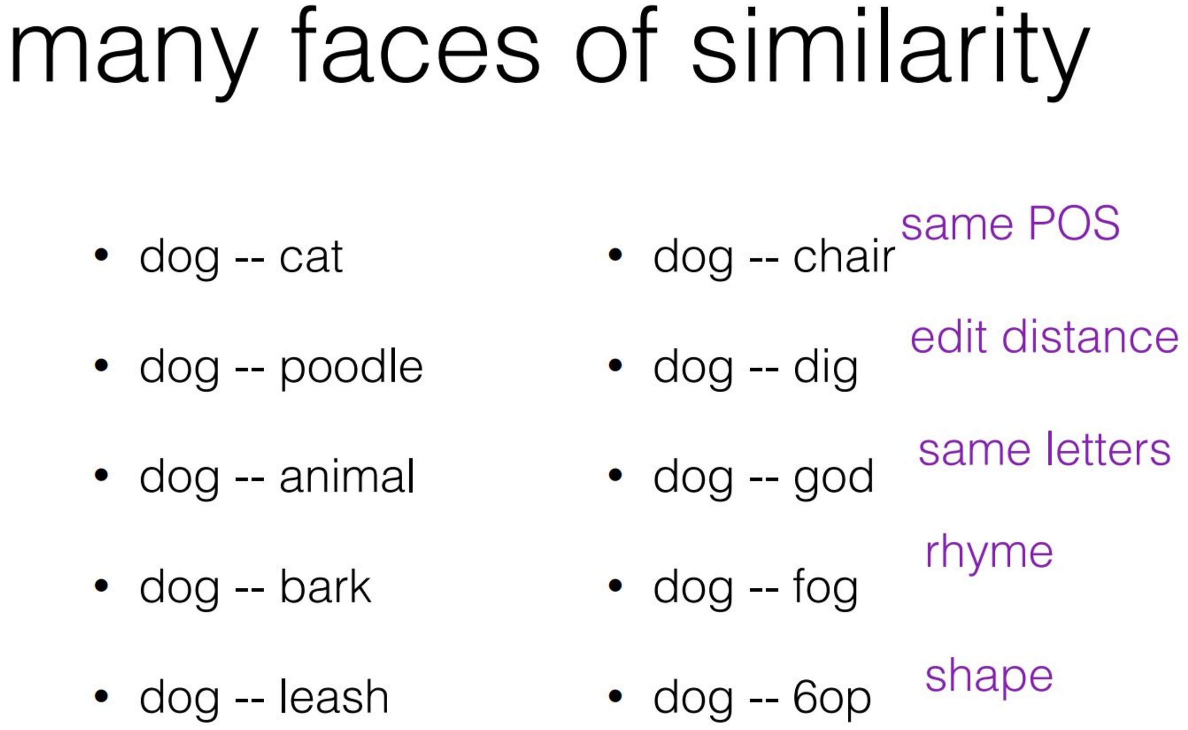
John Rupert Firth -- the originator of the London school of linguistics

Harris, Z. S. (1954). Distributional structure. Word, 10, 146–162. Reprinted in J. Fodor and J. Katz, The Structure of Language, Prentice Hall, 1964
Z. S. Harris, Papers in Structural and Transformational Linguistics, Reidel, 1970, 775–794

Firth, J. R. (1957). A synopsis of linguistic theory 1930– 1955. In Studies in Linguistic Analysis. Philological Society. Reprinted in Palmer, F. (ed.) 1968. Selected Papers of J. R. Firth. Longman, Harlow

REMINDER

What IS ‘similarity’?



many faces of similarity

- dog -- cat
- dog -- poodle
- dog -- animal
- dog -- bark
- dog -- leash

- dog -- chair same POS
- dog -- dig edit distance
- dog -- god same letters
- dog -- fog rhyme
- dog -- 6op shape

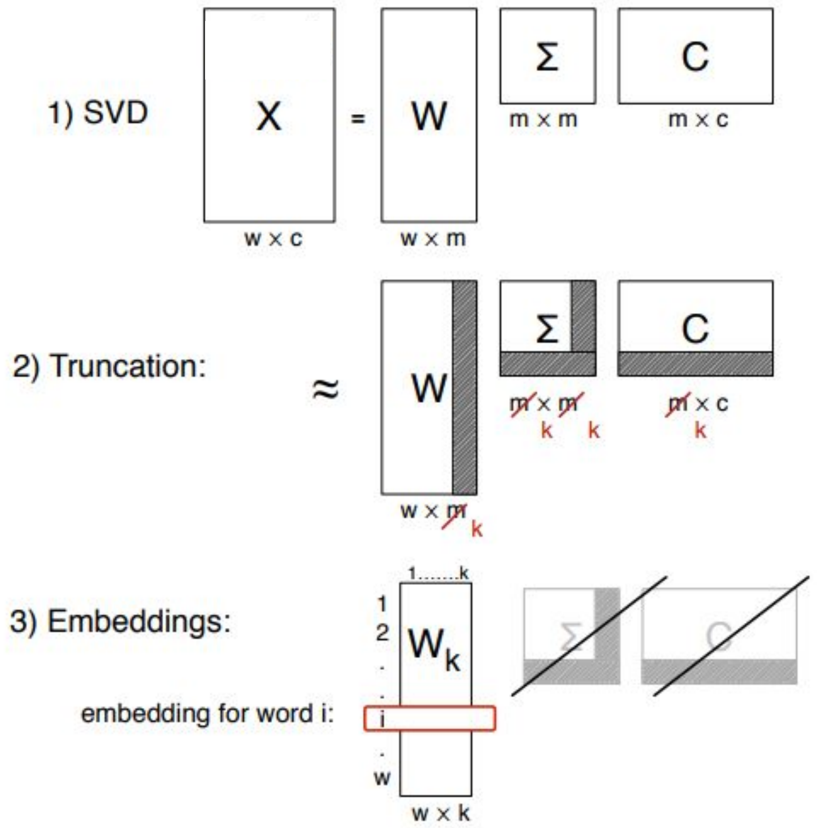
https://speakerdeck.com/mlreview/yoav-goldberg-word-embeddings-what-how-and-whither

REMINDER

Truncated SVD

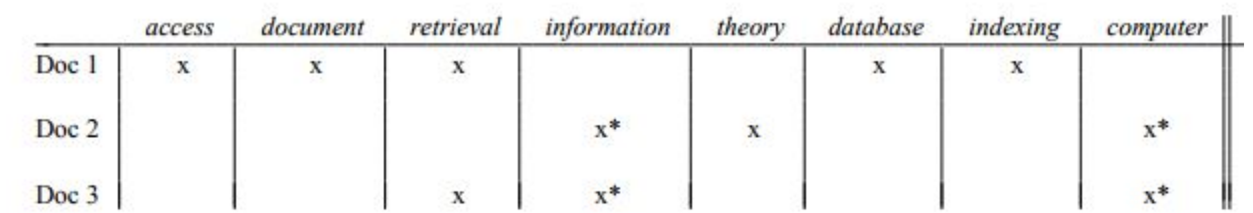
Letting only top K dimensions live

Then our word vector representations are corresponding rows in matrix W_k , that is, k-dimensional vectors



REMINDER

LSA: Latent Semantic Analysis



	access	document	retrieval	information	theory	database	indexing	computer
Doc 1	x	x	x			x	x	
Doc 2				x*	x			x*
Doc 3			x	x*				x*

Applying SVD (**m** = hundreds) to term-document matrix, setting weights as a product of:

the local weight

$$\log f(i,j) + 1$$

the global weight

$$1 + \frac{\sum_j p(i,j) \log p(i,j)}{\log D}$$

for all terms **i** in all documents **j**

S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman. 1988. Using latent semantic analysis to improve access to textual information. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '88), J. J. O'Hare (Ed.). ACM, New York, NY, USA, 281-285.

REMINDER

Truncated SVD for term-term PPMI matrix

We simply apply SVD to word-context matrix and cut off some of the dimensions, choosing **k** manually. Sometimes works better than the sparse analogue.

Other notes on SVD as a way of obtaining vector representations of words:

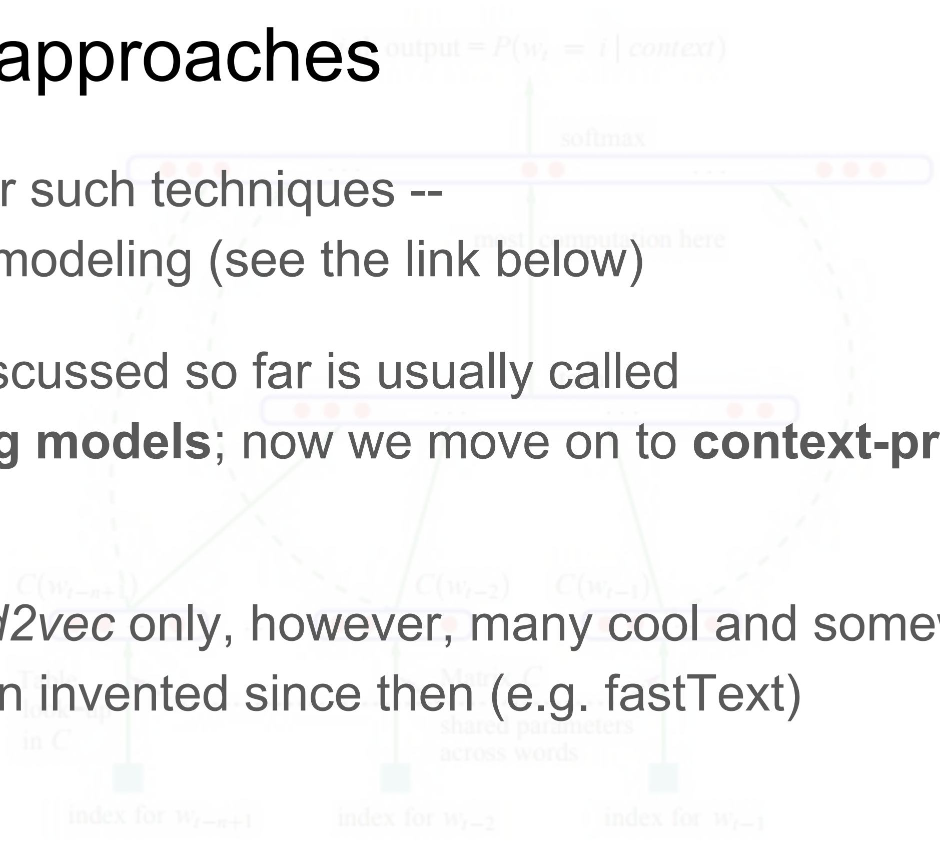
- $(W\Sigma)^T$ can also be treated and used as word vectors (it doesn't work, though)
- Truncating (you never know, but it seems so) helps to generalize and filter out useless information,
- Sometimes throwing away the **first few dimensions** may be helpful

However, it is computationally hard

Lecture plan

1. ~~Sparse vectors~~
    a. ~~“Term-document” approach~~
    b. ~~“Term-term” approach~~
        i. ~~Construction~~
        ii. ~~HAL~~
    c. ~~Weighting~~
    d. ~~Semantic similarity estimation~~
    e. ~~Quality evaluation~~
2. ~~Dense vectors~~
    a. ~~Matrix decomposition~~
    b. “Predictive” approaches

‘Predictive’ approaches



The inspiration for such techniques --
neural language modeling (see the link below)

What we have discussed so far is usually called **context-counting models**; now we move on to **context-predicting models**

We’ll look at *word2vec* only, however, many cool and somewhat similar models have been invented since then (e.g. fastText)

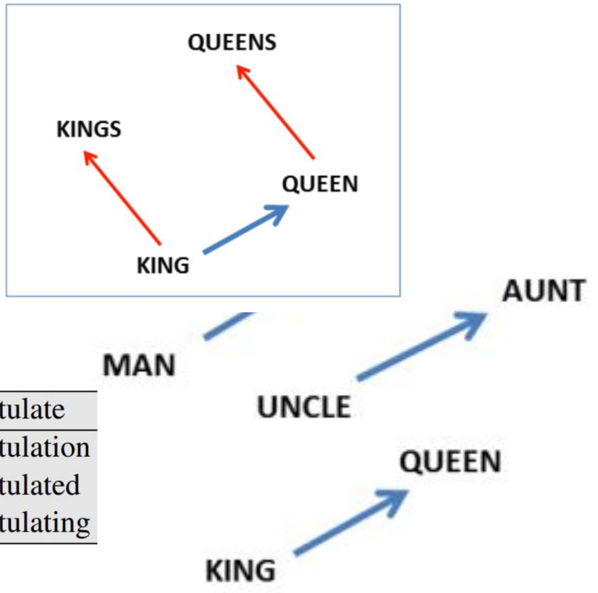
Yoshua Bengio, Réjean Ducharme, Pascal Vincent, Christian Jauvin A Neural Probabilistic Language Model, JMLR 3(Feb):1137-1155, 2003.

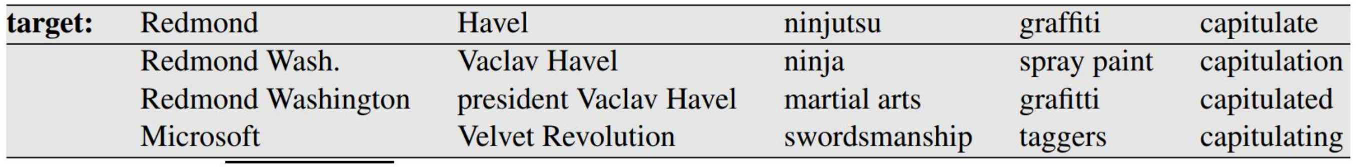
Let’s grumble

2013. Google’s researchers team publishes a paper describing a novel word vectors representations training algorithm, demonstrating that vectors

1) allow to estimate words similarity reasonably well
2) preserve some **relations** as vector subtraction

Thus, thanks to Google’s PR-machine all the coders (even without any linguistic background or interest) around the world now know what distributional semantics is :)





target:	Redmond	Havel	ninjutsu	graffiti	capitulate
	Redmond Wash.	Vaclav Havel	ninja	spray paint	capitulation
	Redmond Washington	president Vaclav Havel	martial arts	grafitti	capitulated
	Microsoft	Velvet Revolution	swordsmanship	taggers	capitulating

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space // In Proceedings of Workshop at ICLR, 2013
Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations // In Proceedings of NAACL HLT, 2013

word2vec is a family of algorithms

SGNS: Skip-grams with Negative Sampling

predicting ‘window contexts’ given the word

CBOW: Continuous Bag-of-Words

predicting the word given the ‘window context’ (won’t discuss)

inb4 -- T. Mikolov:

Skip-gram: works well with small amount of the training data, represents well even rare words or phrases.

CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words

https://www.quora.com/What-are-the-continuous-bag-of-words-and-skip-gram-architectures

skip-grams

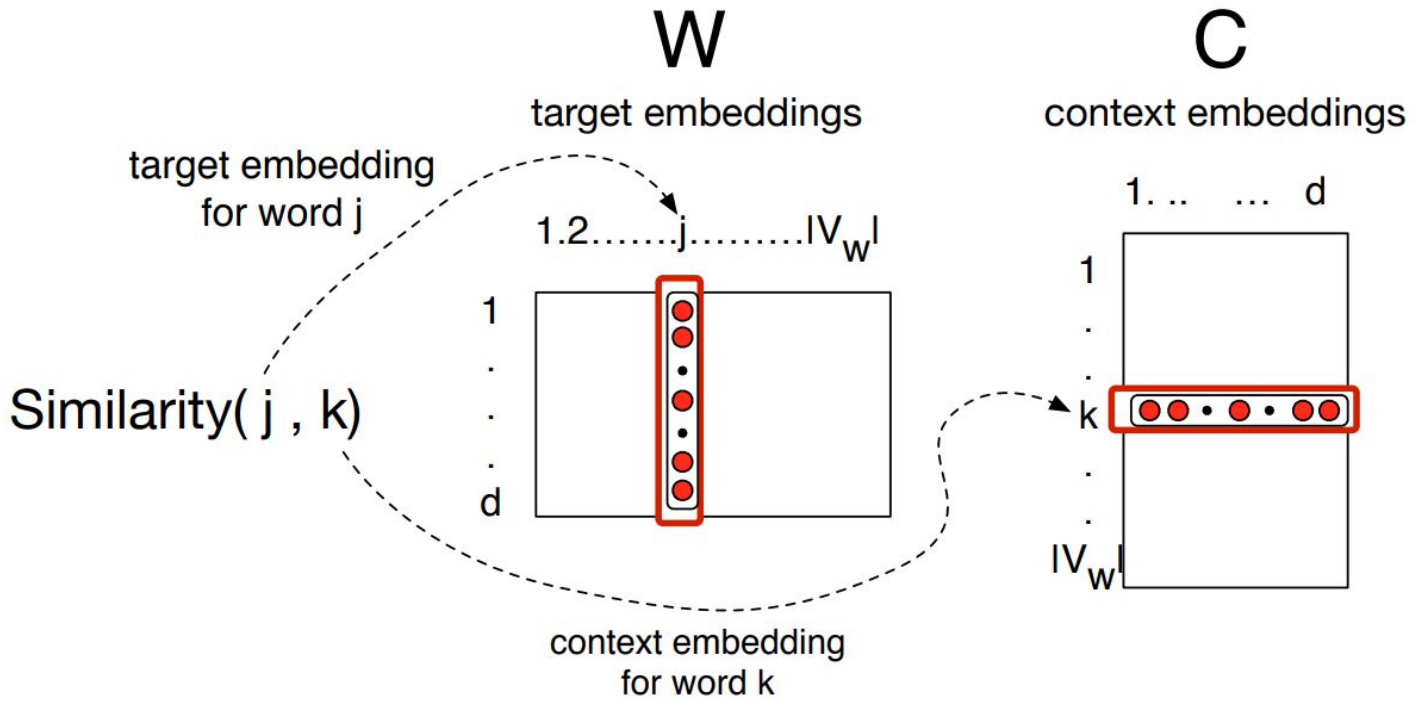
Scanning the text with **2L**-word window and learning to predict context words for the current word; that is, given the word $\mathbf{w_t}$ we estimate the probabilities of its occurrence close to the words $\mathbf{w_{t-L}w_{t-L+1}...w_{t-1}w_{t+1}}...\mathbf{w_{t+L}}$.

Prediction - then correction **based on divergence from true values** -
- prediction - correction - …

Core steps:

1) Each word and each context are paired with a dense vector (initially a random one)
2) Word and context similarity score -- their vectors' scalar product
3) We train vectors values so that $\mathbf{p(v_{context}|v_{word})}$ (computed based on scalar product (2)) for correct contexts were larger

skip-grams


W
target embeddings
C
context embeddings
target embedding for word j
1.2…….j………|V_w|
1
.
.
.
d
1. ..   … d
1
.
.
k
.
.
|V_w|
Similarity(j , k)
context embedding for word k

skip-grams

We’ve measured similarity with cosine distance before and we know it can be treated as ‘normalized scalar product’; we want a similar thing here:

$$\text{Similarity}(j,k) \propto c_k \cdot v_j$$

...but we need probabilities. Then **softmax** is for us

$$p(w_k|w_j) = \frac{exp(c_k \cdot v_j)}{\sum_{i\in|V|} exp(c_i \cdot v_j)}$$

BTW, a problem: a sum of |V| scalar products in the denominator (time-consuming!)
Can be solved with **negative sampling** or **hierarchical softmax**

skip-grams with negative sampling

Computing one probability with **|V|m** multiplication and **|V|(m - 1)** addition ops, and computing **|V|+1** exponent function values is way too expensive

Things can be simplified:

1. maximization of scalar products sigmoids with the **true contexts**,
2. minimization if scalar products sigmoids with **random contexts** (this is what is called here **negative samples**)

$$\sigma(x) = \frac{1}{1+e^{x}}$$

skip-grams with negative sampling

$\sigma(x) = \frac{1}{1+e^x}$

Let’s say we have window of size 2 -- ‘positive’ contexts

```
lemon,  a [tablespoon of apricot preserves or] jam
            c1          c2    w      c3        c4
```

we want to increase this

$$\sigma(c1 \cdot w) + \sigma(c2 \cdot w) + \sigma(c3 \cdot w) + \sigma(c4 \cdot w)$$

k = 2 means the fraction of ‘negative’ contexts is 1:2

```
[cement metaphysical dear coaxial
 n1     n2           n3   n4

apricot attendant whence forever puddle]
        n5        n6     n7      n8
```

we want to decrease this

$$\sigma(n1 \cdot w) + \sigma(n2 \cdot w) + ... + \sigma(n8 \cdot w)$$

skip-grams with negative sampling

Let’s write down the error for every word-context pair

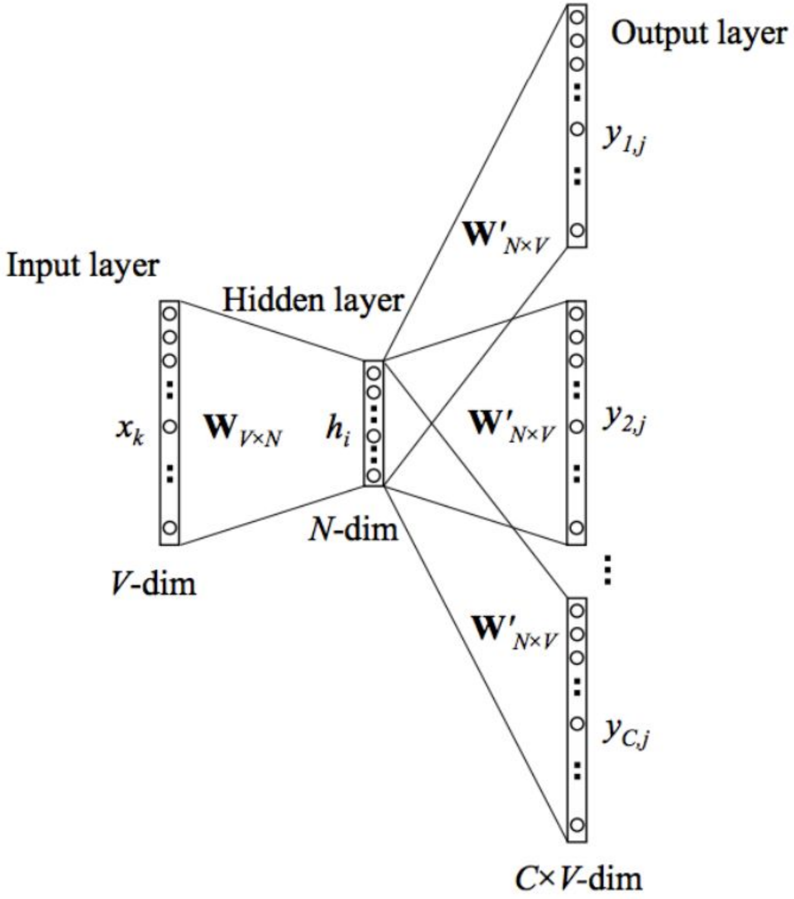
$$\log \sigma(c \cdot w) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim p(w)} \left[\log \sigma(-w_i \cdot w)\right]$$

This is not a SoftMax, but it works

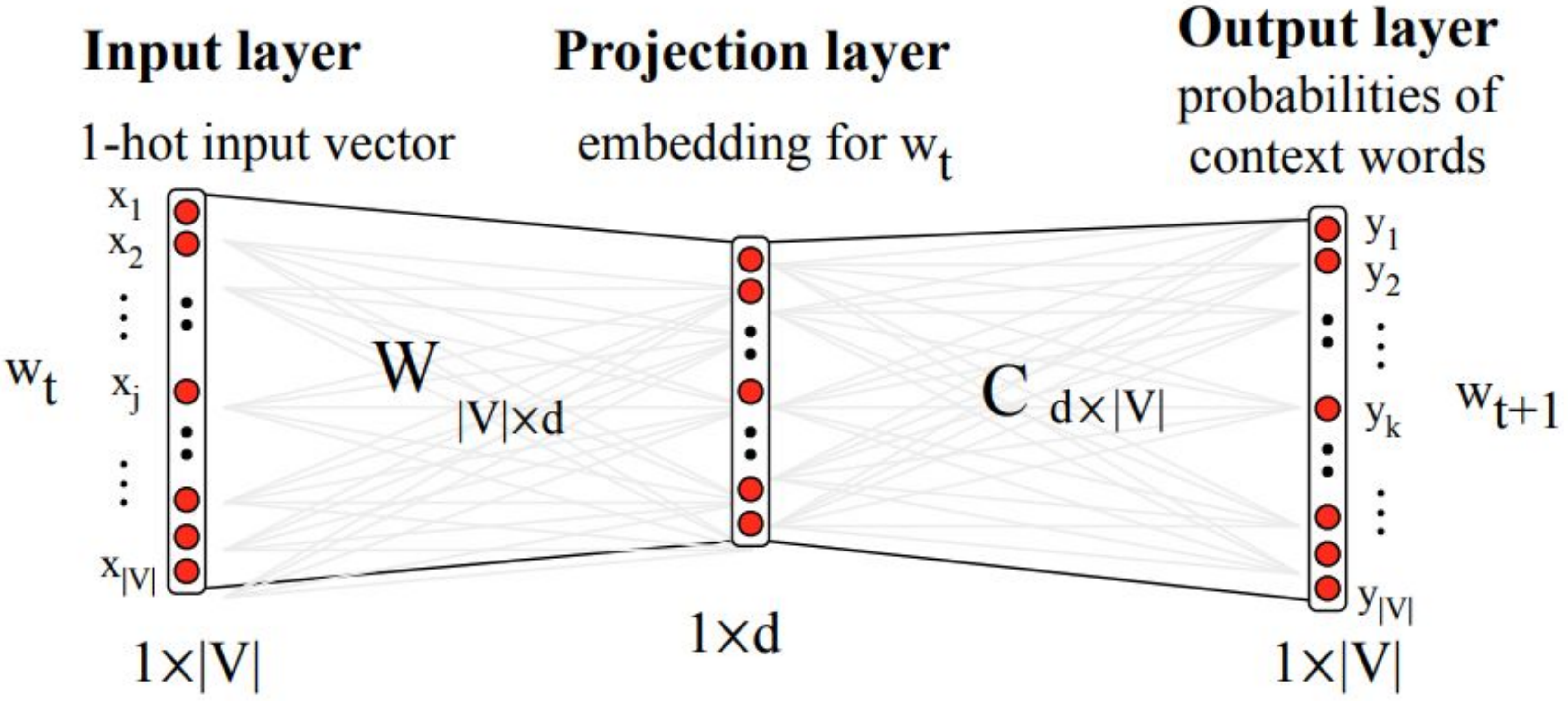
Neural network-like view

Training with backpropagation (BackProp)

(see tutorial или one more)



Neural network-like view



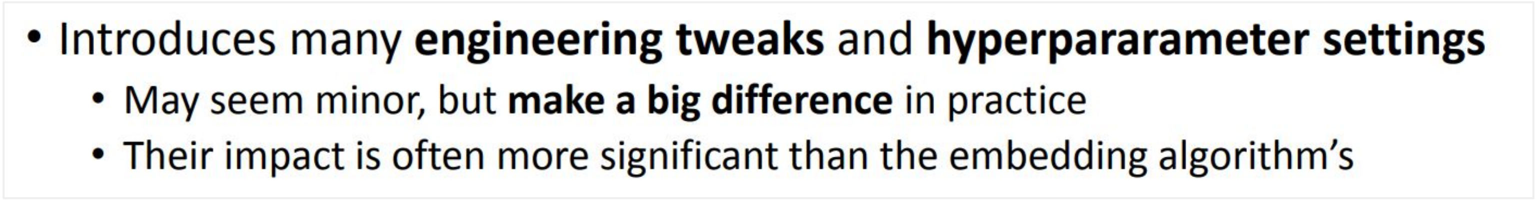
Connection with matrix factorization

It is proved that when skip-gram reaches optimumn the following holds:

$$WC = X^{\text{PMI}} - \log k$$

Which implies that **word2vec** is an implicit matrix factorization of the sparse PMI word-context matrix!

But still it works better. Why?



- Introduces many **engineering tweaks** and **hyperpararameter settings**
  - May seem minor, but **make a big difference** in practice
  - Their impact is often more significant than the embedding algorithm’s

Levy, O. and Goldberg, Y. Neural word embedding as implicit matrix factorization. In NIPS 14, pp. 2177– 2185.

Tools

Many open implementations, mainstream ones are

- **gensim**
- **word2vec** (от Google)
- GloVE (Stanford)
- fastText (FacebookAIResearch)
- implementations in popular NN frameworks

Pretrained vectors for different languages, e.g.

- RusVectores
- Not sure if this list is complete and/or good
  (however, you can always google vectors for your language of interest)

Datasets

For English

WordSim-353 - 353 noun pairs with ‘similarity scores’ estimates from 0 to 10
SimLex-999 - similar task with different parts-of-speech + synonymy is important
TOEFL dataset - 80 quizzes: a word + four more, the task is to choose a synonym
Also there are datasets where contexts are also available

For Russian

Translations of standard datasets + thesauri data
https://github.com/nlpub/russe-evaluation

Also see

Other popular vector representations

Glove: J. Pennington, R. Socher, C. Manning. Global Vectors for Word Representation EMNLP2014
fastText: P. Bojanowski, E.Grave, A. Joulin,T. Mikolov. Enriching word vectors with subword information, 2016.

Text representations

doc2vec: Le Q., Mikolov T. Distributed representations of sentences and documents // ICML-14

Handling polysemy:

AdaGram: S. Bartunov, D. Kondrashkin, A. Osokin, D. Vetrov. Breaking Sticks and Ambiguities with Adaptive Skip-gram. International Conference on Artificial Intelligence and Statistics (AISTATS) 2016.

And many more...

Used/recommended materials

1. Martin/Jurafsky, Ch. 15
2. Yoav Goldberg: word embeddings what, how and whither
3. Papers on slides
4. Valentin Malykh from ODS/iPavlov on w2v
5. A very cool explanation of what word2vec is
6. Wikipedia

Vector semantics -III

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg

NRU ITMO, St Petersburg, 2018
anton.m.alexeyev+itmo@gmail.com

Many thanks to Denis Kirjanov for words of advice