Unsupervised NLP

Vsevolod Dyomkin

prj-nlp, 2019-04-25

Potential

* lots of unlabeled data
  or data with weak supervision
* not everything may be labeled
* examples:
  - web
  - books
  - parallel corpora for MT

Unsupervised Approaches

* counting
* matrix factorization
* expectation maximization
* clustering

Word Relations

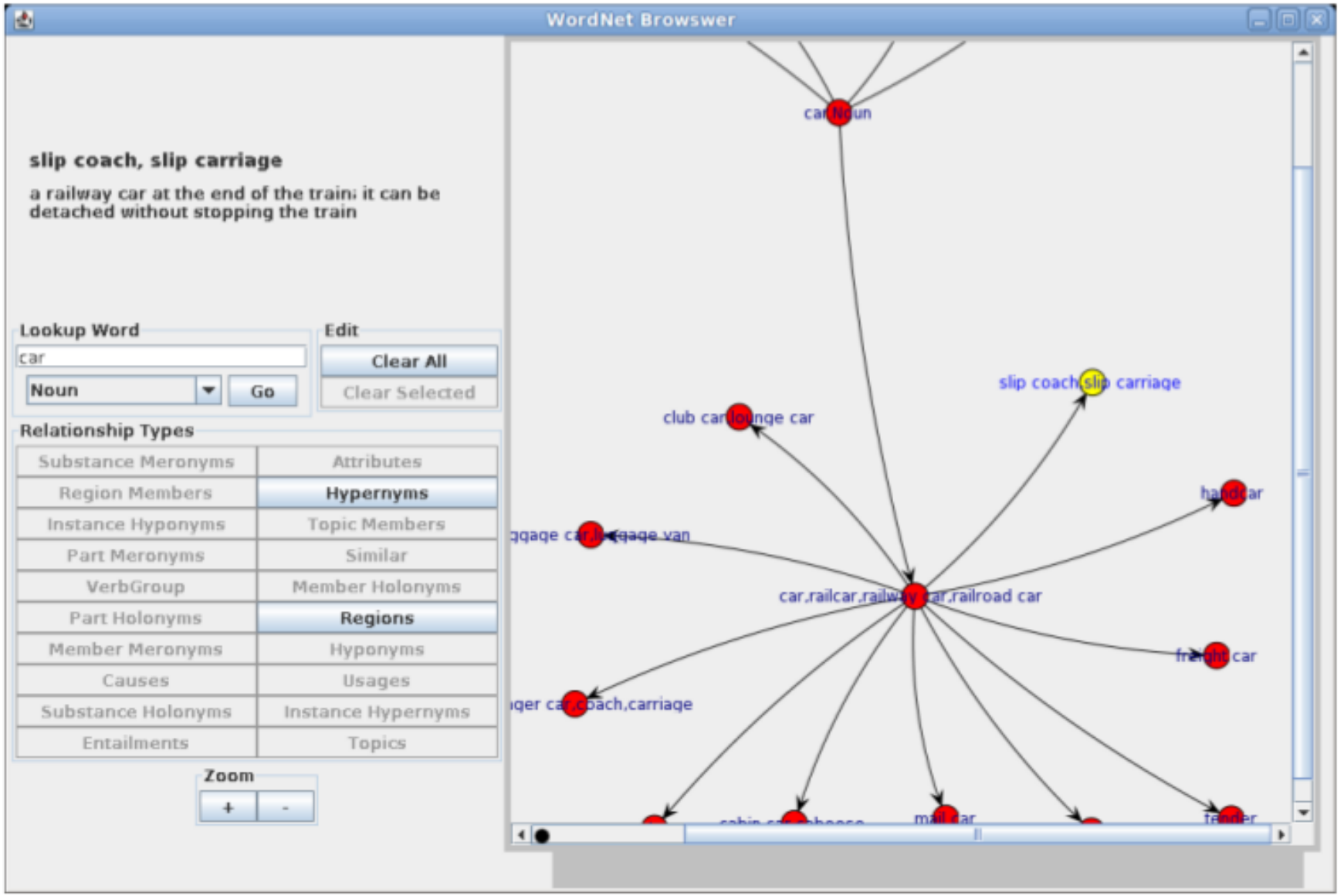
Question 1: how are words related?
Question 2: how to measure word similarity?

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

Graph-based Approach



Wordnet

Graph-based Word Similarity

$$Sim(C1, C2) = 2 \bullet Max(C1, C2) - SP$$

$$Sim_{Rod}(C1^p, C2^q) = W_w S_w(C1^p, C2^q) + W_u S_u(C1^p, C2^q) + W_n S_n(C1^p, C2^q))$$

$$Sim_{Resnik}(C1, C2) = \frac{2 \bullet \ln((p_{mis}(C1,C2))}{\ln(p(c1)) + \ln(p(c2))}$$

$$Sim_{Knappe}(C1, C2) = p * \frac{|Ans(C1) \cap Ans(C2)|}{|Ans(C1)|} + (1-p) * \frac{|Ans(C1) \cap Ans(C2)|}{|Ans(C2)|}$$

$$Sim_{Zhou}(C1, C2) = 1 - k\left(\frac{\ln(len(C1,C2)+1}{\ln(2 \bullet (deep_{max}-1))}\right) - (1-k) * ((IC(C1) + IC(C2) - 2 * IC(lso(C1, C2))/2)$$

$$Sim_{Resnik}(C1, C2) = -ln(p_{mis}(C1,C2))$$

$$Sim_{tvsk}(C1, C2) = \frac{|C1 \cap C2|}{|C1 \cap C2| + \propto |C1 - C2| + (\propto - 1)|C2 - C1|}$$

$$Sim_{LC}(C1, C2) = -\log\left(\frac{length}{2.D}\right)$$

$$Sim_{HSO}(C1, C2) = C - SP - k \bullet d$$

$$Sim_{wup}(C1, C2) = \frac{2 \bullet N}{N1 + N2 + 2 \bullet N}$$

Graph-based Word Similarity

$$Sim(C1, C2) = 2 \bullet Max(C1, C2) - SP$$

$$Sim_{Rod}(C1^p, C2^q) = W_w S_w(C1^p, C2^q) + W_u S_u(C1^p, C2^q) + W_n S_n(C1^p, C2^q))$$

$$Sim_{Resnik}(C1, C2) = \frac{2 \bullet \ln((p_{mis}(C1,C2))}{\ln(p(c1)) + \ln(p(c2))}$$

Sim_K

Doesn't work!

$\frac{\backslash Ans(C2)|}{(C2)|}$

$$Sim_{Zhou}(C1, C2) = 1 - \kappa \left(\frac{}{\ln(2 \bullet (deep_{max} - 1))}\right) -$$

$$(1 - k) * ((IC(C1) + IC(C2) - 2 * IC(lso(C1, C2))/2)$$

$$Sim_{Resnik}(C1, C2) = -ln(p_{mis}(C1,C2))$$

$$Sim_{tvsk}(C1, C2) = \frac{|C1 \cap C2|}{|C1 \cap C2| + \propto |C1 - C2| + (\propto - 1)|C2 - C1|}$$

$$Sim_{LC}(C1, C2) = -\log\left(\frac{length}{2.D}\right)$$

$$Sim_{HSO}(C1, C2) = C - SP - k \bullet d$$

$$Sim_{wup}(C1, C2) = \frac{2 \bullet N}{N1 + N2 + 2 \bullet N}$$

https://arxiv.org/pdf/1310.8059.pdf

Distributional Hypothesis

"You shall know a word by the company it keeps"
--John Rupert Firth

Explicit word representation - number of nonzero dimensions:

- max:474234
- min:3
- mean:1595
- median:415

Co-occurance Matrix

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Pointwise Mutual Information (PMI)

Dan Jurafsky

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*} p_{*j}}$$

	p(w,context)					p(w)
	computer	data	pinch	result	sugar	
apricot	0.00	0.00	0.05	0.00	0.05	0.11
pineapple	0.00	0.00	0.05	0.00	0.05	0.11
digital	0.11	0.05	0.00	0.05	0.00	0.21
information	0.05	0.32	0.00	0.21	0.00	0.58
p(context)	0.16	0.37	0.11	0.26	0.11	

- pmi(information,data) = $\log_2$ (.32 / (.37*.58)) = .58

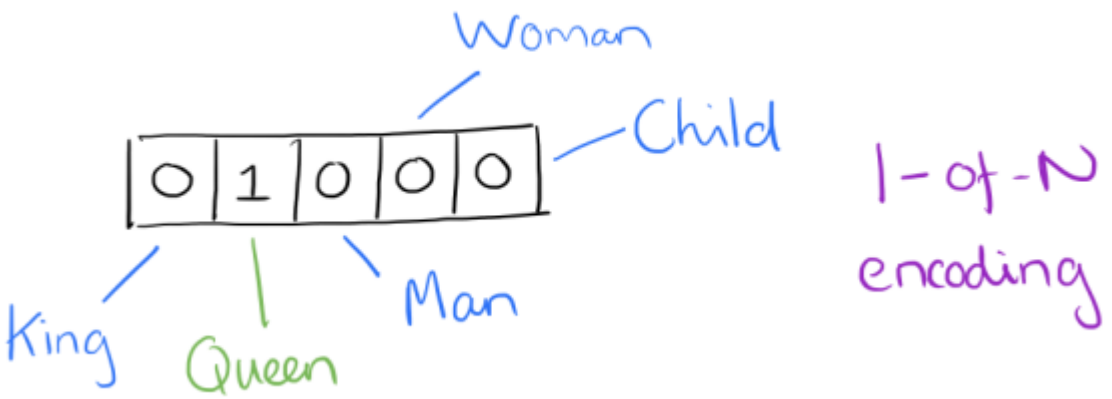
(.57 using full precision)

PPMI(w,context)

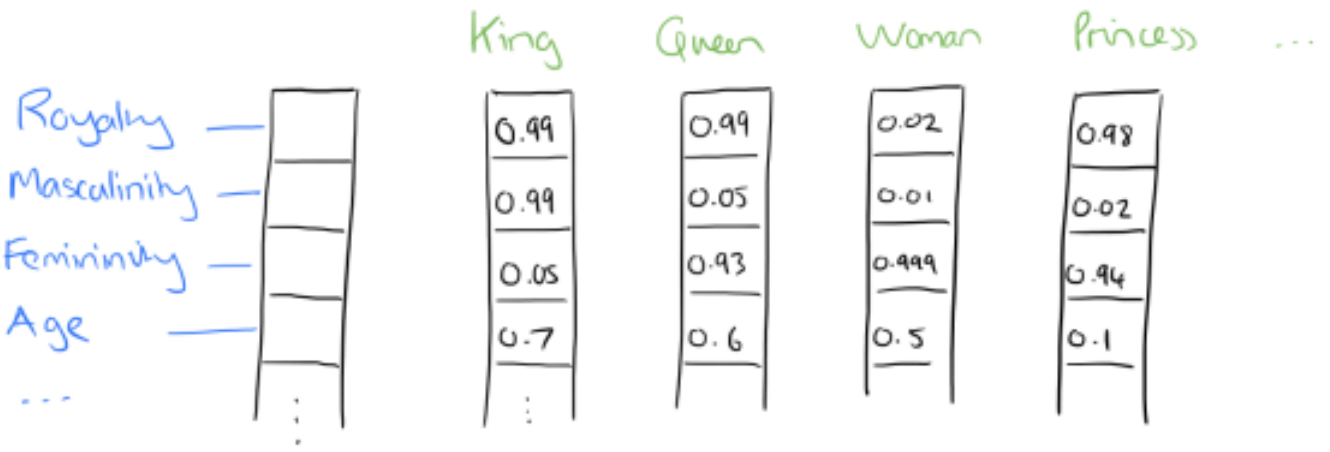
	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

Word Vector Embeddings

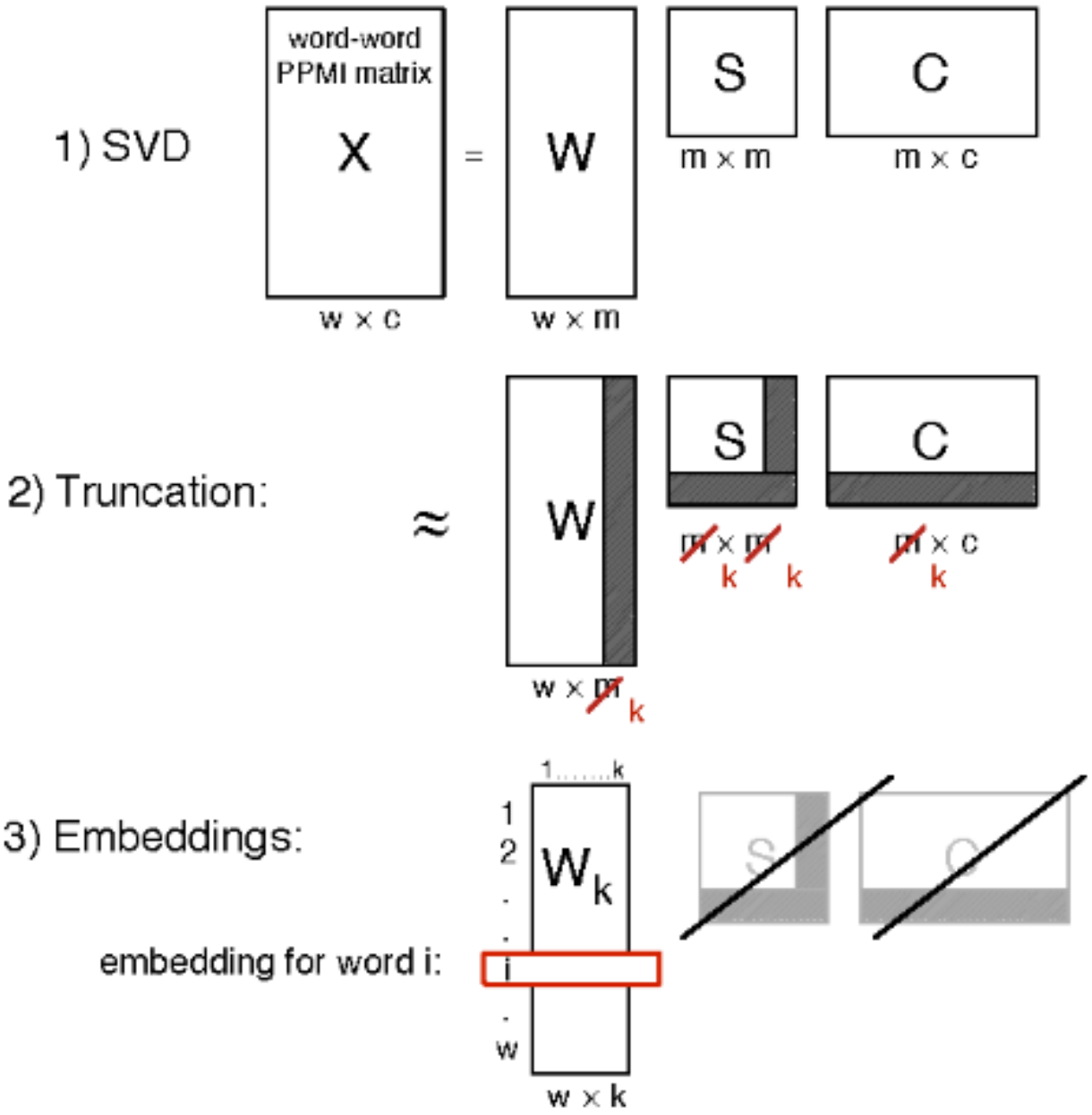
* 1-hot



* sparse
* dense



SVD


1) SVD
word-word
PPMI matrix
X
=
W
S
C
w × c
w × m
m × m
m × c
2) Truncation:
≈
W
S
C
w × m
k
m × m
k
k
m × c
k
3) Embeddings:
1.....k
1
2
W_k
embedding for word i:
i
w
w × k

NNSE

Non-Negative Sparse Embedding
- using non-negative matrix factorization
- and sparse coding
http://talukdar.net/papers/nnse_coling12.pdf

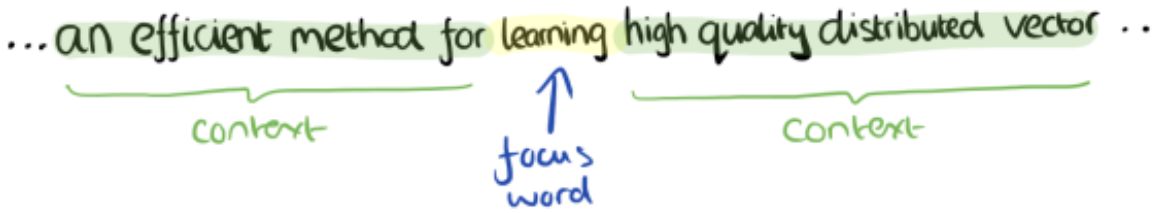
$$C(\mathbf{A}, \mathbf{S}) = \frac{1}{2}\|\mathbf{X} - \mathbf{AS}\|^2 + \lambda \sum_{ij} f(S_{ij}), \quad (1)$$

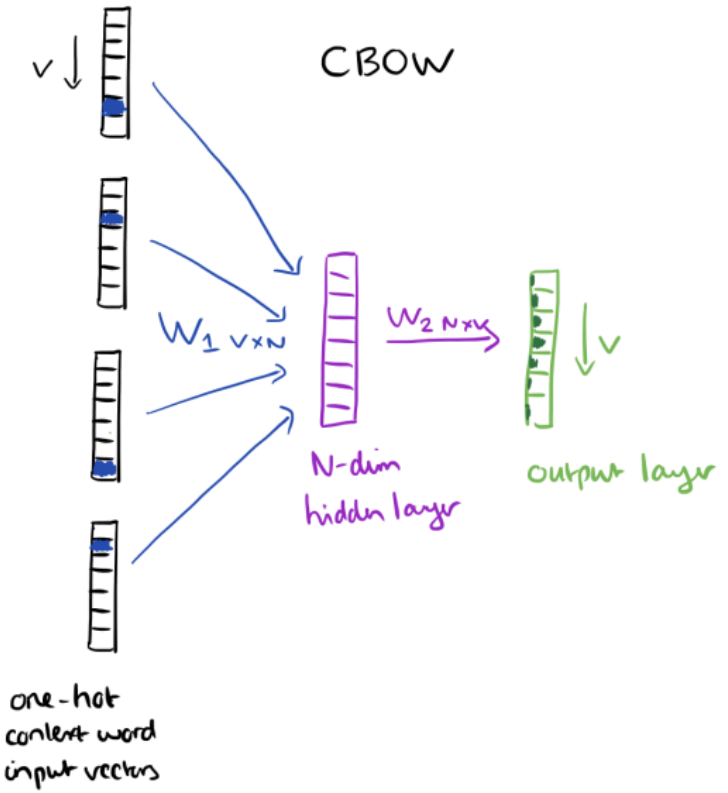
where the squared matrix norm is simply the summed squared value of the elements, i.e. $\|\mathbf{X}-\mathbf{AS}\|^2 = \sum_{ij}[\mathbf{X}_{ij}-(\mathbf{AS})_{ij}]^2$. The tradeoff between sparseness and accurate reconstruction is controlled by the parameter λ, whereas the form of f defines how sparseness is measured. To achieve a sparse code, the form of f must be chosen correctly: A typical choice is $f(s) = |s|$, although often similar functions that exhibit smoother behaviour at zero are chosen for numerical stability.

CNNSE (Compositional):
- add composition constraint to training
http://www.aclweb.org/anthology/N15-1004

word2vec

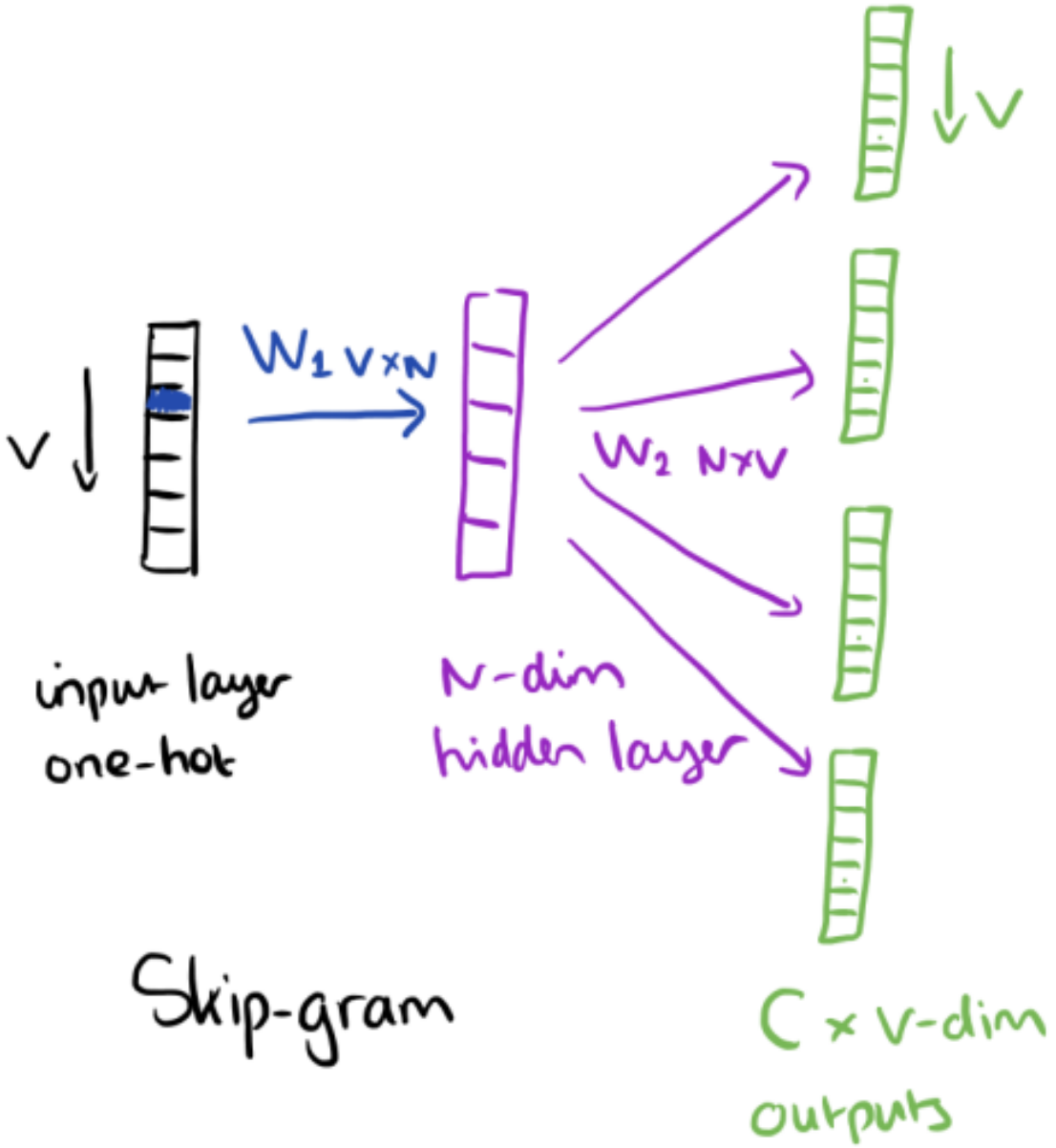
Continuous bag-of-words (CBOW)





http://u.cs.biu.ac.il/~yogo/cvsc2015.pdf

word2vec - SGNS



Expectation Maximization (EM)

Other uses:

* alignment
* WSD
* clustering
* estimation of parameters for NB, HMM, GMM models

Word Vectors Evaluation

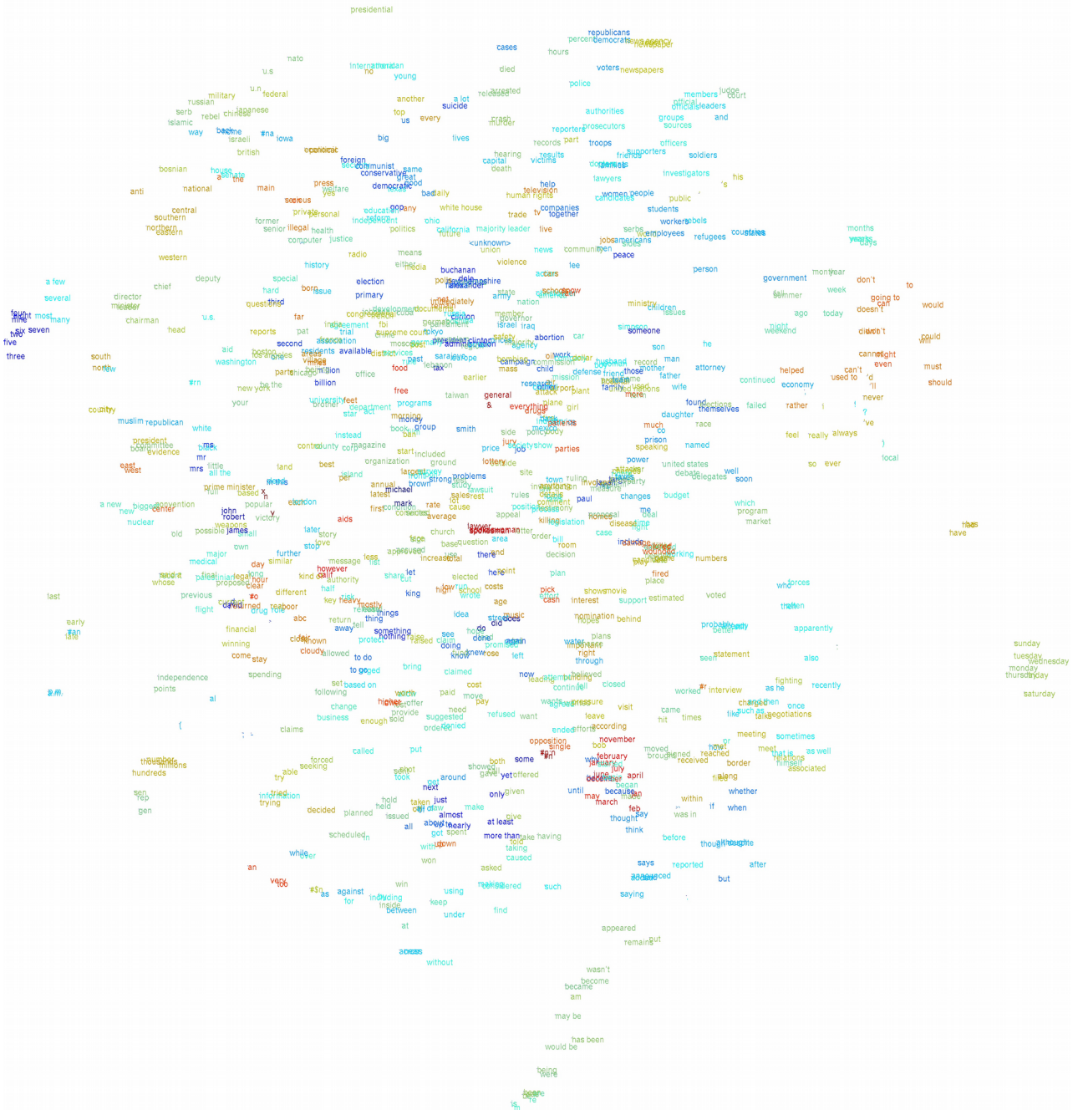
* extrinsic
* intrinsic
  - relatedness
  - analogy
  - categorization
  - selectional preference

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

https://aclanthology.info/pdf/D/D15/D15-1036.pdf

Visualization — t-SNE

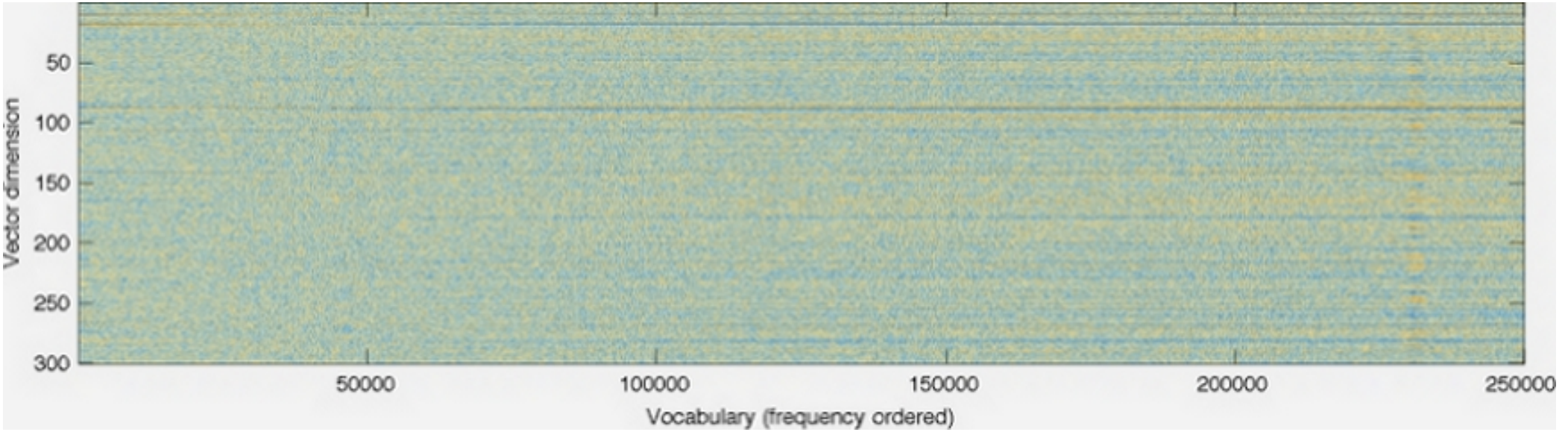


Problems of Word Vectors

* OOVs
* polysemy
* low-frequency words
* non-explicit dimensions

GloVe

* explicit matrix factorization



fasttext

Extension to SGNS to take into account subword information (character ngrams):

The word “where” is reprsented as a sum of representations of “<where>”, “<wh”, “whe”, “her”, “ere”, “re>”

https://arxiv.org/pdf/1607.04606.pdf

ConceptNet Numberbatch

~~Wordnet~~ConceptNet strikes back

The current SOTA vectors due to:
* vector ensemble using ConceptNet to merge vectors
* OOV handling

https://blog.conceptnet.io/2016/05/25/conceptnet-numberbatch-a-new-name-for-the-best-wOrd-embeddings-you-can-download/

https://blog.conceptnet.io/2017/03/02/how-luminoso-made-conceptnet-into-the-best-word-vectors-and-won-at-semeval/

word2gauss

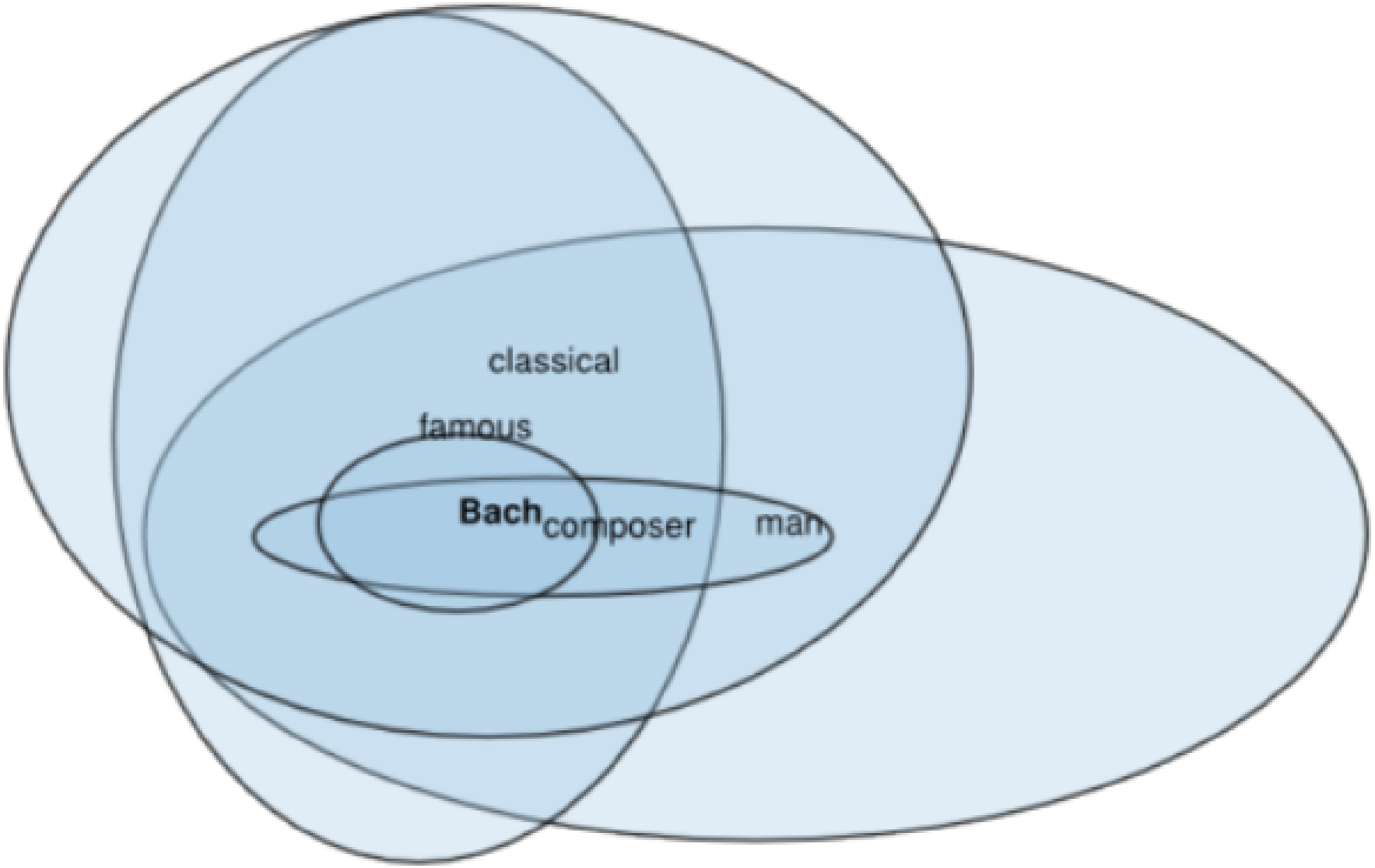
Each word is represented as a multivariate Gaussian: a probability P[i] — a K-dimensional Gaussian parameterized by mean mu and co-variance matrix Sigma:

$$P[i] \sim N(x; \mu[i], \Sigma[i])$$

The mean is a vector of length K and in the most general case Sigma[i] is a (K, K) matrix. 2 approximations to simplify Sigma:
- diagonal - a vector length K
- spherical - a float

https://arxiv.org/pdf/1412.6623.pdf
https://github.com/seomoz/word2gauss

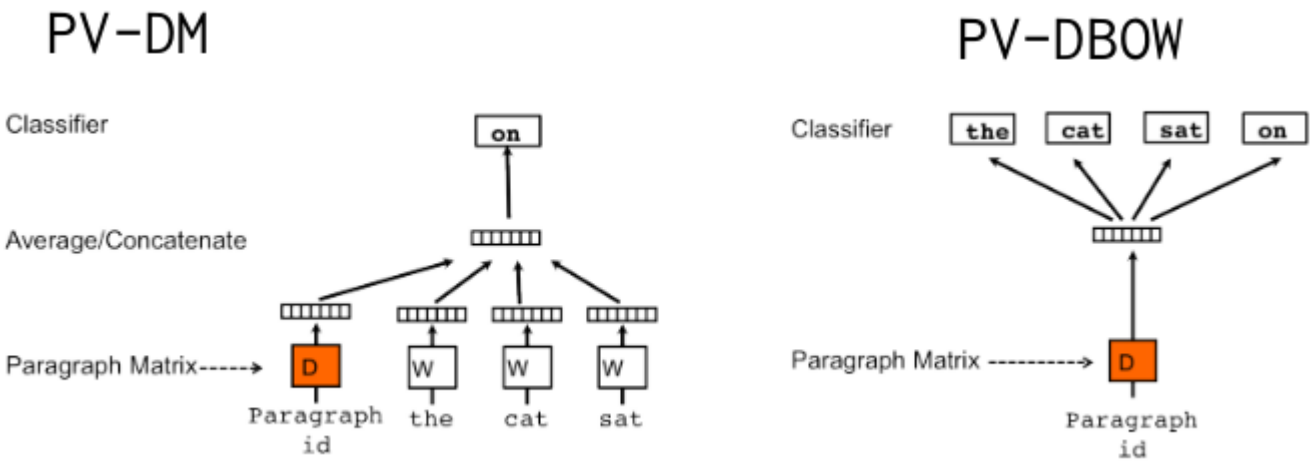
word2gauss



Document Embeddings

Question: how to represent phrases/sentences/paragraps/documents with dense vectors?

Default answer: average the word vectors
Alternative: “paragraph vectors”



https://cs.stanford.edu/~quocle/paragraph_vector.pdf

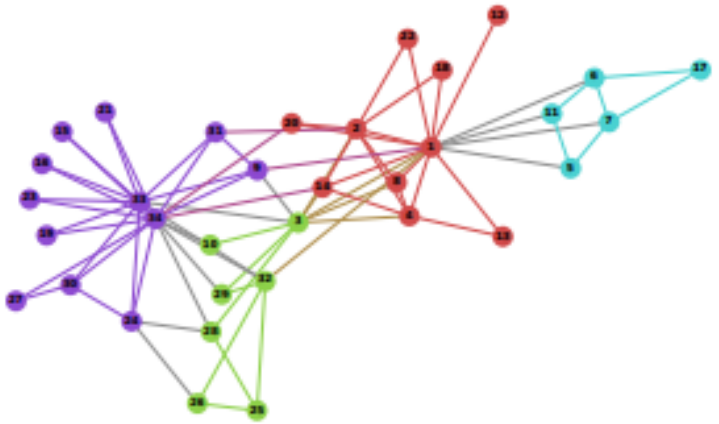
Other Approaches

* Skip-thoughts
* Universal Sentence Encoder
* ElMo

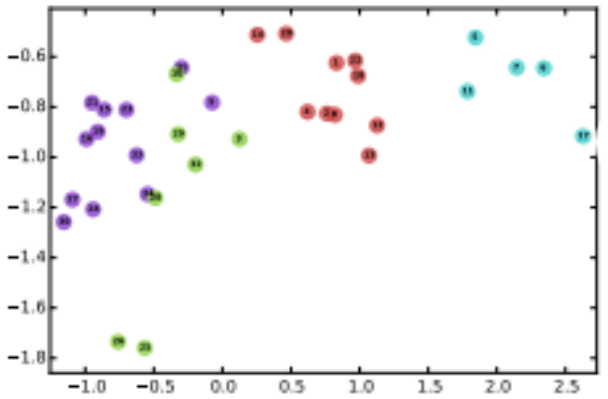
Graph Embeddings

They exist! :)

DeepWalk algorithm



(a) Input: Karate Graph



(b) Output: Representation

https://arxiv.org/pdf/1403.6652.pdf

Dense Representations Recap

Key idea: transition from sparse (BoW) to dense vectors and maximize the vectors' affinity to some relation in the process.

Pros:
- capture those relations
- easier to compute with (possible to use as input for neural nets)

Cons:
- expensive to compute the vectors themselves
- knowledge transfer

Topic Modelling

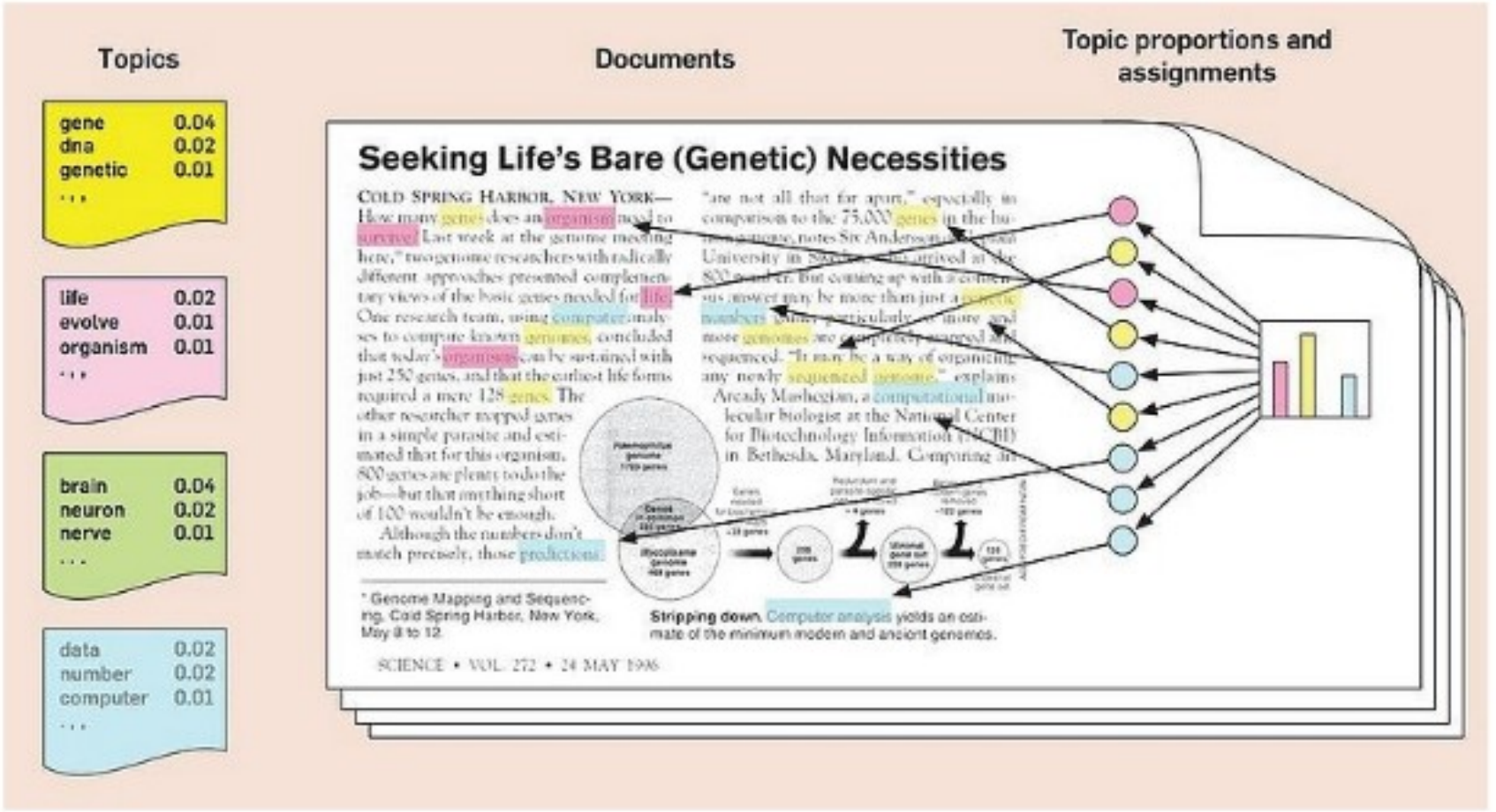
A multi-class whole-text classification/ranking problem.

A mostly **unsupervised** problem.

Latent Semantic Indexing

Factorization of the word-document matrix using SVD and selecting the top-N eigen values.

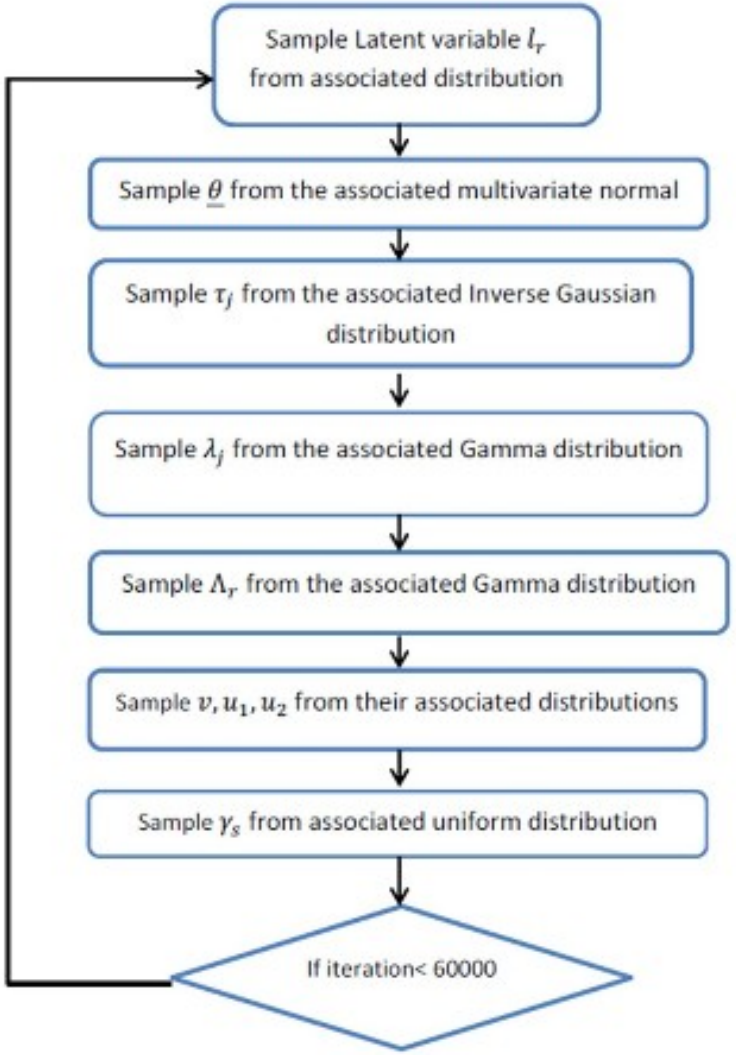
Latent Dirichlet Allocation



http://www.cl.cam.ac.uk/teaching/1213/L101/clark_lectures/lect7.pdf

Gibbs Sampling

Used to estimate LDA models.



https://stats.stackexchange.com/questions/10213/can-someone-explain-gibbs-sampling-in-very-simple-words

Anchor Words

Problem of LSI/LDA: hard to interpret topics.

Alternative factorization to SVD: Non-negative matrix factorization (NMF).

https://cs.stanford.edu/~rishig/courses/ref/l9b.pdf

Read More

Vector representations:
https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/
https://arxiv.org/pdf/1411.2738v3.pdf
https://blog.acolyer.org/2016/06/01/distributed-representations-of-sentences-and-documents/
https://github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/doc2vec-IMDB.ipynb
https://arxiv.org/pdf/1607.05368.pdf
https://arxiv.org/pdf/1411.4166.pdf
https://arxiv.org/pdf/1403.6652.pdf

Topic models:
http://pages.cs.wisc.edu/~jerryzhu/cs769/latent.pdf
https://www.youtube.com/watch?v=3mHy4OSyRf0
https://cs.stanford.edu/~rishig/courses/ref/l9b.pdf
https://www.quora.com/What-is-an-intuitive-explanation-of-the-Dirichlet-distribution