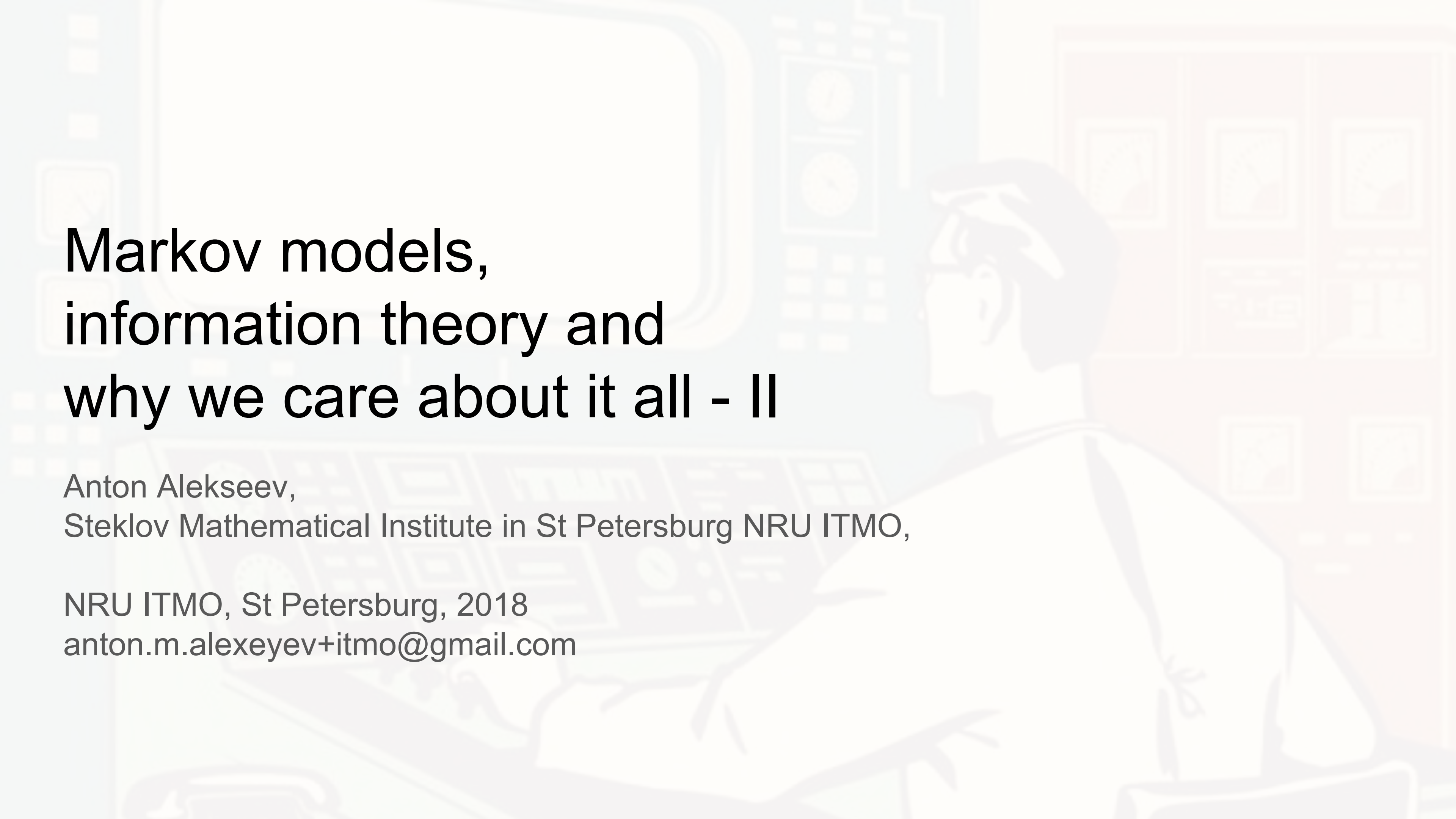


Markov models, information theory and why we care about it all - II

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg NRU ITMO,

NRU ITMO, St Petersburg, 2018

anton.m.alexeyev+itmo@gmail.com

Markov chains

REMINDER

- Evident enough, probability of trajectory of length n is computed like that

$$p(x_a, ..., x_z) = \pi_a \prod_{i=2}^{|steps|} P_{steps[i],steps[i+1]}, steps = (a, ..., z)$$

- It is easy to prove that the vector of probabilities of the process to be in certain states at $m-$th step can be computed like that

$$\pi^{(m)} = (p_1^{(m)}, ..., p_n^{(m)}) = \pi P_{tr}^m$$

Markov chains: the limit

REMINDER

One can demonstrate that if $P_{trans\ i,j} = p_{i \to j} > 0$, there exist a single asymptotic distribution

$$\hat{\mathbf{p}} = \lim_{m \to \infty} \pi P^m_{trans},$$

and

$$\hat{\mathbf{p}} = \hat{\mathbf{p}} P_{trans}, \sum \hat{p}_i = 1$$

Such distribution is called the **stationary** one.

Stationary distribution: interpretation

REMINDER

Suppose we are watching random [web] surfer, who moves from state to state **eternally**, making decisions where to glide using the distribution of states in the current row

http://slideplayer.com/slide/8080871/

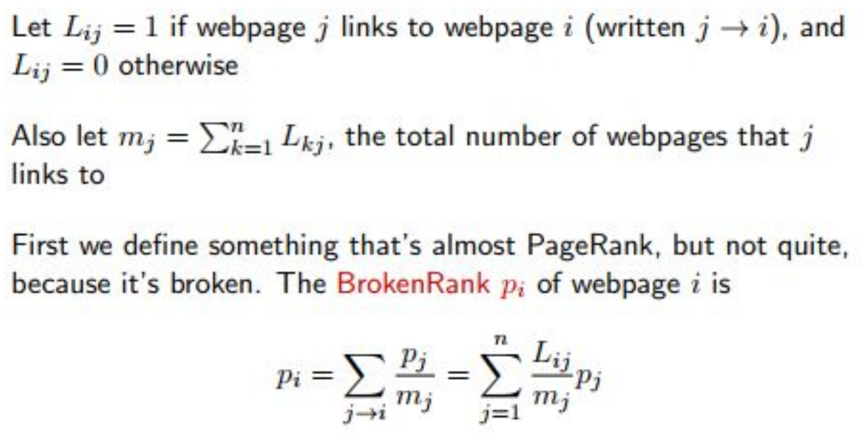
Then each value in the vector of stationary distribution is **the fraction of total time** spent in the corresponding state

Application example №2: PageRank

REMINDER

The ‘value’ of the web page is defined by

- the ‘value’ of the pages that refer to it,
- a number of pages those pages refer to (less = better)



Let $L_{ij} = 1$ if webpage j links to webpage i (written $j \to i$), and $L_{ij} = 0$ otherwise

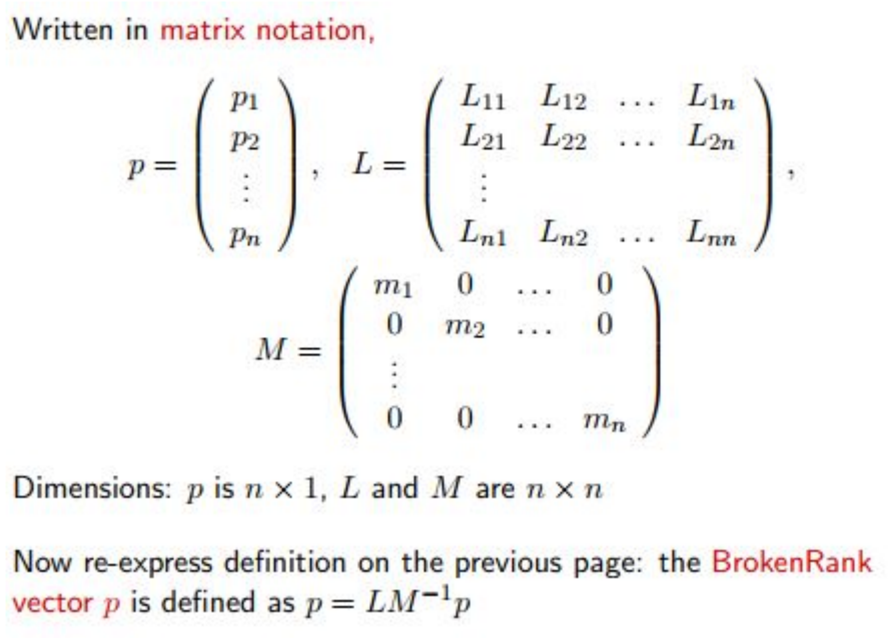
Also let $m_j = \sum_{k=1}^{n} L_{kj}$, the total number of webpages that j links to

First we define something that's almost PageRank, but not quite, because it's broken. The BrokenRank p_i of webpage i is

$$p_i = \sum_{j \to i} \frac{p_j}{m_j} = \sum_{j=1}^{n} \frac{L_{ij}}{m_j} p_j$$

http://www.stat.cmu.edu/~ryantibs/datamining/lectures/03-pr.pdf

Application example №2: PageRank



Written in matrix notation,

$$p = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{pmatrix}, \quad L = \begin{pmatrix} L_{11} & L_{12} & \dots & L_{1n} \\ L_{21} & L_{22} & \dots & L_{2n} \\ \vdots \\ L_{n1} & L_{n2} & \dots & L_{nn} \end{pmatrix},$$

$$M = \begin{pmatrix} m_1 & 0 & \dots & 0 \\ 0 & m_2 & \dots & 0 \\ \vdots \\ 0 & 0 & \dots & m_n \end{pmatrix}$$

Dimensions: p is $n \times 1$, L and M are $n \times n$

Now re-express definition on the previous page: the BrokenRank vector p is defined as $p = LM^{-1}p$

Does that remind us of anything? Yep, stationary distribution!

Application example №2: PageRank

$$\mathrm{P}(\text{go from } i \text{ to } j) = \begin{cases} 1/m_i & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$$

Cool!

1. set the probabilities as above,
2. compute the stationary distribution,
3. use it as a quality/value measure,
4. ??????
5. PROFIT

Or not?

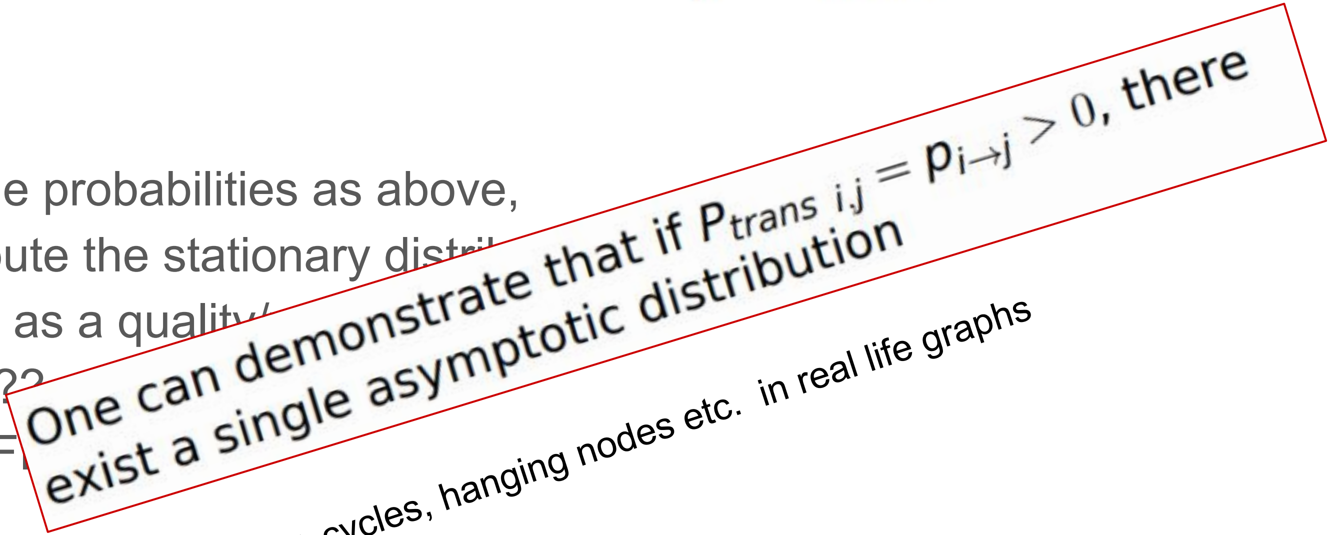
Application example №2: PageRank

$$\mathrm{P}(\text{go from } i \text{ to } j) = \begin{cases} 1/m_i & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$$

Cool!

1. set the probabilities as above,
2. compute the stationary distri
3. use it as a quality/
4. ??????
5. PROF

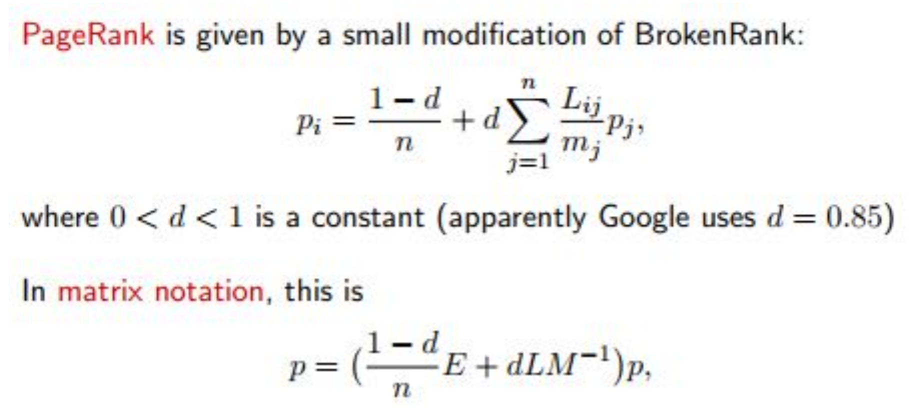
Or not?



One can demonstrate that if $P_{\text{trans } i,j} = p_{i \to j} > 0$, there exist a single asymptotic distribution

+ cycles, hanging nodes etc. in real life graphs

Application example №2: PageRank



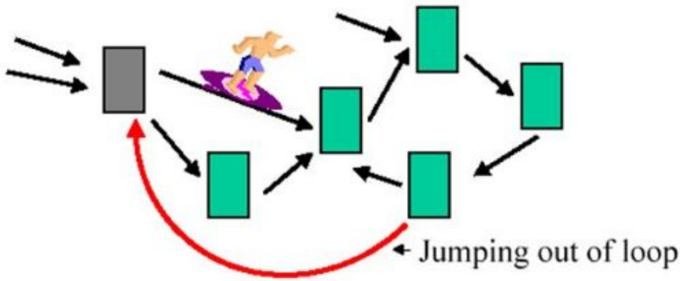
PageRank is given by a small modification of BrokenRank:

$$p_i = \frac{1-d}{n} + d\sum_{j=1}^{n} \frac{L_{ij}}{m_j} p_j,$$

where $0 < d < 1$ is a constant (apparently Google uses $d = 0.85$)

In matrix notation, this is

$$p = (\frac{1-d}{n}E + dLM^{-1})p,$$



Which means that once in a while, e.g. 15 times out of 100, we allow our surfer to jump to a completely random page

$$\mathrm{P}(\text{go from } i \text{ to } j) = \begin{cases} (1-d)/n + d/m_i & \text{if } i \to j \\ (1-d)/n & \text{otherwise} \end{cases}$$

Actually Google owe their success to a completely different algorithm https://archive.google.com/pigeonrank/

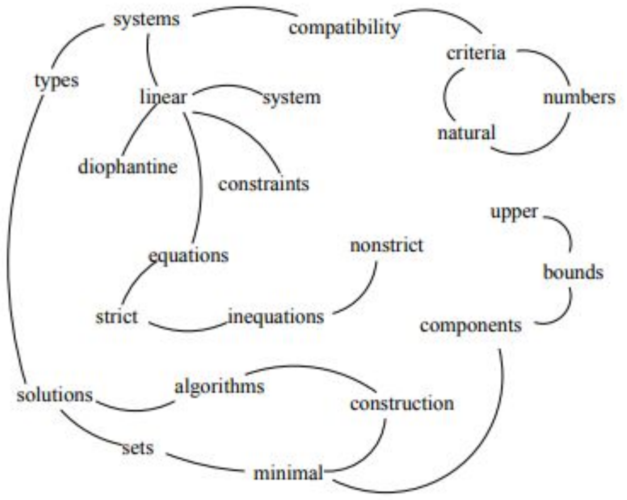
Application example №3: PageRank (TextRank)

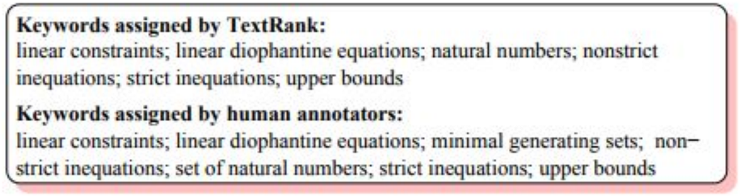
General idea:

- text as a graph
- textual entity (word/sentence/...) having MAX PageRank is the most important one

E.g. keywords:

1) tokenize text,
2) filter out words by part-of-speech,
3) a graph: if the number of words between a pair of words is greater than N, draw an edge between them
4) compute PageRank,
5) merge the close nodes with high PageRank into one ("Matlab" -> "code" => "Matlab_code").




Keywords assigned by TextRank:
linear constraints; linear diophantine equations; natural numbers; nonstrict inequations; strict inequations; upper bounds

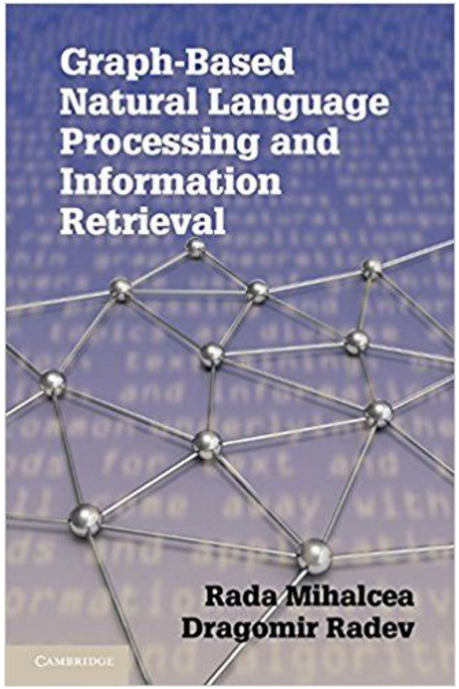
Keywords assigned by human annotators:
linear constraints; linear diophantine equations; minimal generating sets; non-strict inequations; set of natural numbers; strict inequations; upper bounds

Please note

Graph-based NLP is also a way to look at text mining tasks, there is a 2011 book on that:

- graph theory
- probability theory
- linear algebra
- social networks analysis methods
- natural language processing, finally

But we are not going to discuss all that, *of course*


Graph-Based Natural Language Processing and Information Retrieval
Rada Mihalcea
Dragomir Radev
CAMBRIDGE

Other applications

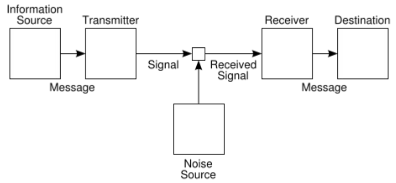
- Language detection
- Named-entity recognition
- POS-tagging
- Speech recognition
- …useful almost every time we deal with sequences

Plan for today: theory and applications

1. ~~Markov chains~~
    a. ~~Language models~~
    b. ~~Keywords extraction and other applications~~

2. Elements of information theory
    a. Information
        i. Collocations extraction
        ii. One weird trick to estimate sentiment
    b. Entropy
        i. Connection between entropy and perplexity

Information theory elements: entropy and C°

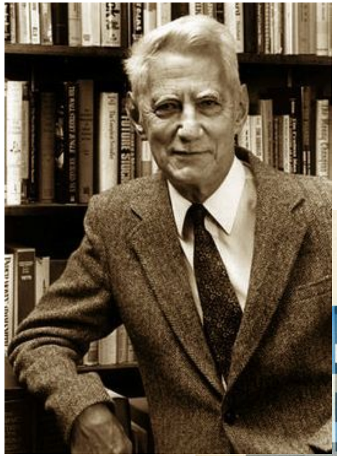


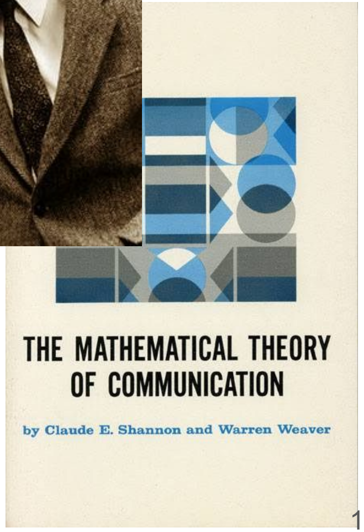
1948 - A Mathematical Theory of Communication,
Claude Shannon; information theory foundations are introduced

1949 - published as a book with Warren Weaver's commentary

Information entropy and bit are introduced

Found applications in compression algorithms, cryptography, signal processing, etc.




THE MATHEMATICAL THEORY OF COMMUNICATION
by Claude E. Shannon and Warren Weaver

Self-Information

- How much information the object represents; the less probable (or the more 'sudden') the event, the greater the information

  $$I(X) = -log_2 p(x)$$

  (log base may be different)
- Example: it is known that the event occurred
  $p(x) = 1$
  Then
  $$I(x) = 0$$
- Uniform distribution: $p(x_i) = \frac{1}{N} \quad \forall x \in 1 : N$

  $$I(x_i) = -log_2 N^{-1} = log_2 N,$$

  length of the binary code of number of values!

Self-Information

- If all words are equally frequent and occur independently, we can't 'compress' the text (we'll have to encode all words with numbers), otherwise

  —

  $p(x_0) = \frac{1}{3}, p(x_1) = \frac{1}{3}, p(x_2) = \frac{1}{3}$

  $$I_0 = log_2 3, I_1 = log_2 3, I_2 = log_2 3$$

  $p(x_0) = \frac{2}{3}, p(x_1) = \frac{1}{6}, p(x_2) = \frac{1}{6}$

  $$I_0 = log_2 3/2, I_1 = log_2 6, I_2 = log_2 6$$

- Rare events are the most 'informative'

  $$=$$

  we can afford to encode them with long codes

Self-Information

- BTW, if $p(x_0) = 0.5, p(x_1) = p_1, ..., p(x_n) = p_n$

  $$I(x_0) = -log_2 0.5 = 1 \ bit$$

  (the name of the measure of information depends on the log base)
- NB! We do not depend on other frequencies distribution!

* Mutual information

A measure of “common volume of information” shared by X and Y

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log\left(\frac{p(x,y)}{p(x)\,p(y)}\right),$$

- When X and Y are independent, equals zero
- When there’s functional dependency, turns into X-s entropy (or Y-s entropy)

Is used, e.g. for feature selection

Pointwise mutual information

PMI

$$\operatorname{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}.$$

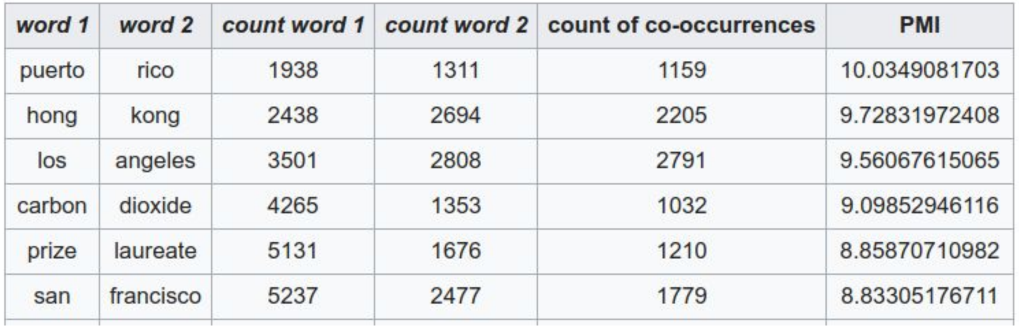
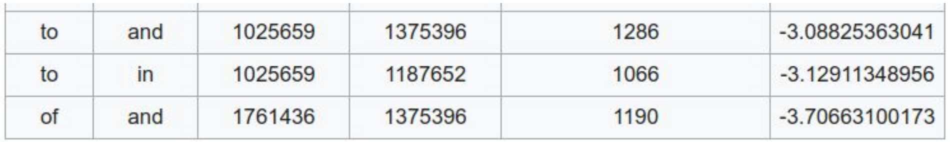
Intuitively:

- PMI shows the volume of added information about word2, when we see the word1
- Can be applied to non-consecutive words in the text
- Gives large weight to rare phrases
- Reasonable to use as a measure of independence, or as a measure of non-randomness of co-occurence (we’ll use this)

Pointwise mutual information: example №1

Collocations extraction: ***if words co-occur a little less frequently than they occur on their own, they are collocations***; probabilities estimated as frequencies

Wikipedia, Oct. 2015

word 1	word 2	count word 1	count word 2	count of co-occurrences	PMI
puerto	rico	1938	1311	1159	10.0349081703
hong	kong	2438	2694	2205	9.72831972408
los	angeles	3501	2808	2791	9.56067615065
carbon	dioxide	4265	1353	1032	9.09852946116
prize	laureate	5131	1676	1210	8.85870710982
san	francisco	5237	2477	1779	8.83305176711
to	and	1025659	1375396	1286	-3.08825363041
to	in	1025659	1187652	1066	-3.12911348956
of	and	1761436	1375396	1190	-3.70663100173

https://en.wikipedia.org/wiki/Pointwise_mutual_information

Pointwise mutual information: example №2

Not a SOTA (lol, 2002 paper), but a smart idea of using web search engines for sentiment analysis:

1. Using POS-aware patterns, extract certain word collocations
2. Query AltaVista:
   "poor", "<extr. phrase> NEAR poor",
   "excellent", "<extr. phrase> NEAR excellent"
3. Compute and average Semantic Orientation for all phrases; if SO > 0 then **positive**

A search operator available in AltaVista

$$\mathrm{PMI}(word_1, word_2) = \log_2\left(\frac{\mathrm{p}(word_1 \,\&\, word_2)}{\mathrm{p}(word_1)\,\mathrm{p}(word_2)}\right)$$

$$\mathrm{SO}(phrase) = \mathrm{PMI}(phrase, \text{“excellent”}) - \mathrm{PMI}(phrase, \text{“poor”})$$

$$\mathrm{SO}(phrase) = \log_2\left[\frac{\mathrm{hits}(phrase \text{ NEAR “excellent”})\ \mathrm{hits}(\text{“poor”})}{\mathrm{hits}(phrase \text{ NEAR “poor”})\ \mathrm{hits}(\text{“excellent”})}\right]$$

Peter D. Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02). Association for Computational Linguistics, Stroudsburg, PA, USA, 417-424.

Plan for today: theory and applications

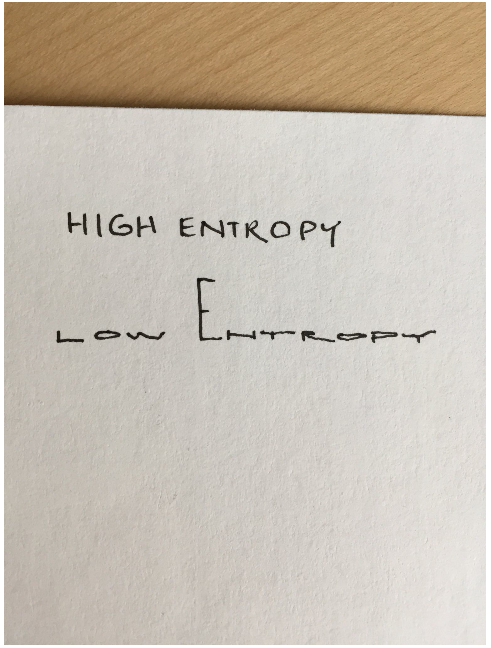
1. ~~Markov chains~~
   a. ~~Language models~~
   b. ~~Keywords extraction and other applications~~

2. ~~Elements of information theory~~
   a. ~~Information~~
      i. ~~Collocations extraction~~
      ii. ~~One weird trick to estimate sentiment~~
   b. Entropy

Literature, recommendations

1. Martin-Jurafsky 3 ed., Chapter 4
2. NLP course @ CSC 2014
3. **PageRank (better explanations and material is more complete)**
   Anand Rajaraman and Jeffrey David Ullman. 2011.
   Mining of Massive Datasets
4. Ryan Tibshirani, Data Mining lectures slides
5. Wikipedia + relevant materials links on it
6. Романовский И. В. Дискретный анализ: Учебное пособие для студентов, специализирующихся по прикладной математике и информатике

Information entropy



https://twitter.com/dmimno/status/968856022164148224

Information entropy

$$H(X) = -\sum_{x \in X} p(x) log_2 p(x),$$

X — «predicted values»
Possible interpretations:

- self-information expected value (as a measure of «meaningfulness»),
- a measure of «unpredictability» of the system $\mathbb{E}_{p_X} I(X)$,
- ...

Information entropy

- Entropy — is the only function (up to a constant factor) that has the following properties:
  1. continuity
  2. symmetry
     (the reordering of probabilities changes nothing)
  3. maximal for uniform distribution
  4. given that the distribution is uniform, outcomes number increase implies entropy increase

$$H_N(\frac{1}{N}, ..., \frac{1}{N}) < H_{N+1}(\frac{1}{N+1}, ..., \frac{1}{N+1})$$

  5. grouping outcomes leads to losing information the following way:

$$H_N(\frac{1}{N}, ..., \frac{1}{N}) = H_k(\frac{b_1}{N}, ..., \frac{b_k}{N}) + \sum_{i=1}^{k} \frac{b_i}{N} H(\frac{1}{b_i}, ..., \frac{1}{b_i}),$$

$$b_1 + ... + b_k = N$$

- Proved by C. Shannon.

Cross entropy

Cross entropy — average number of bits necessary for recognition of the event if the coding scheme is based on the given probability distribution q instead of the 'true' p. *«Wikipedia»*

$$H(p, q) = -\sum_{i=1}^{n} p(x_i) log_2 q(x_i)$$

We use the 'true' distribution for weighting the estimates information.

Cross entropy and her friends

$$H(p,q) = -\sum_{i=1}^{n} p(x_i) log_2 q(x_i) + H(p) - H(p) =$$

$$= \sum_{i=1}^{n} p(x_i)(log_2 p(x_i) - log_2 q(x_i)) + H(p) = D_{KL}(p||q) + H(p)$$

- D_{KL} — Kullback-Leibler divergence
- **VERY IMPORTANT**

$$H(p) \leq H(p,q)\ \forall p,q$$

which is why cross entropy is useful: the more precise is the estimate q, the smaller the difference + cross entropy will never overestimate the 'true' entropy

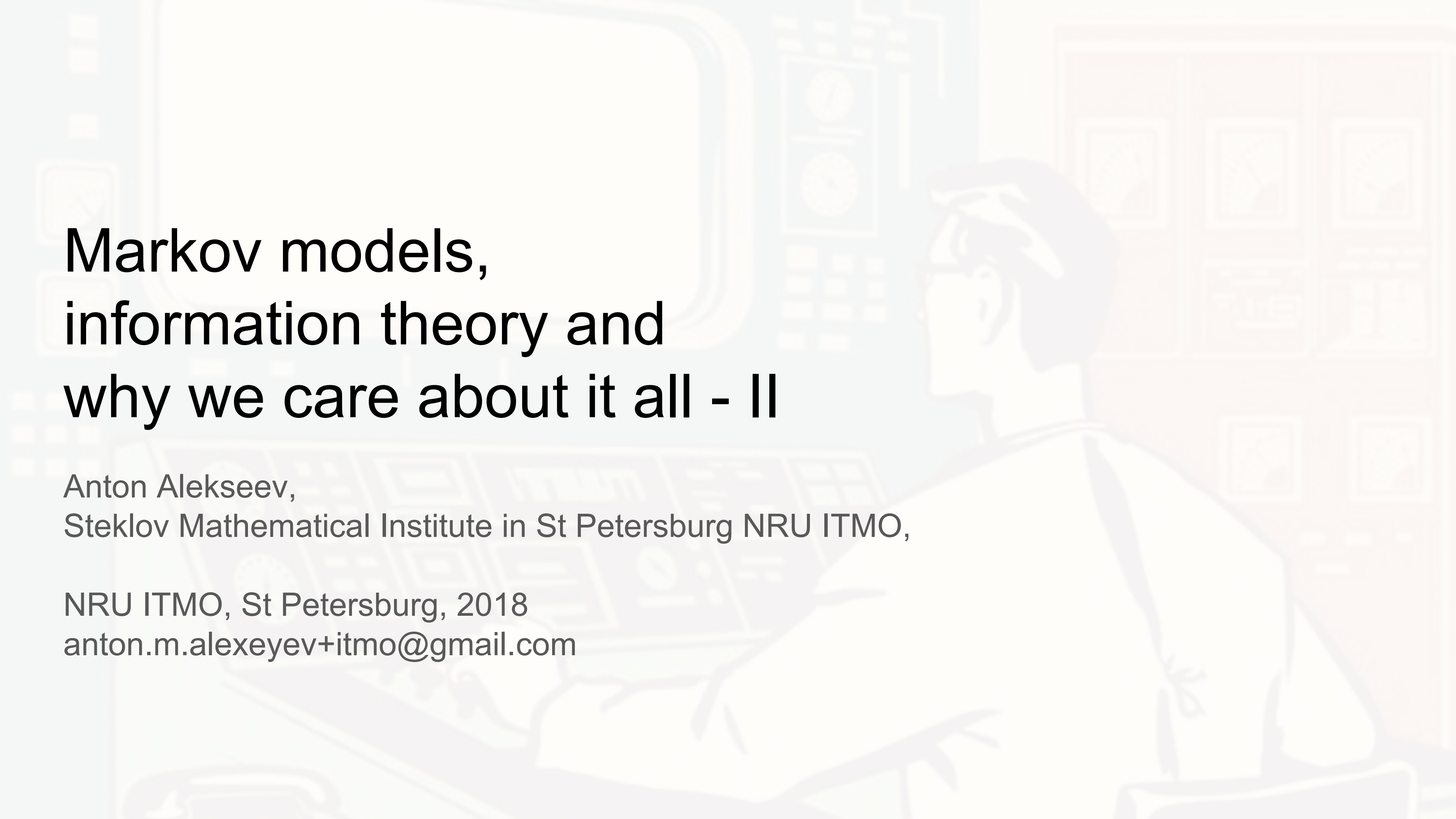
BTW*

An interesting point of view on mutual information

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log\left(\frac{p(x,y)}{p(x)\,p(y)}\right),$$

$$\Updownarrow$$

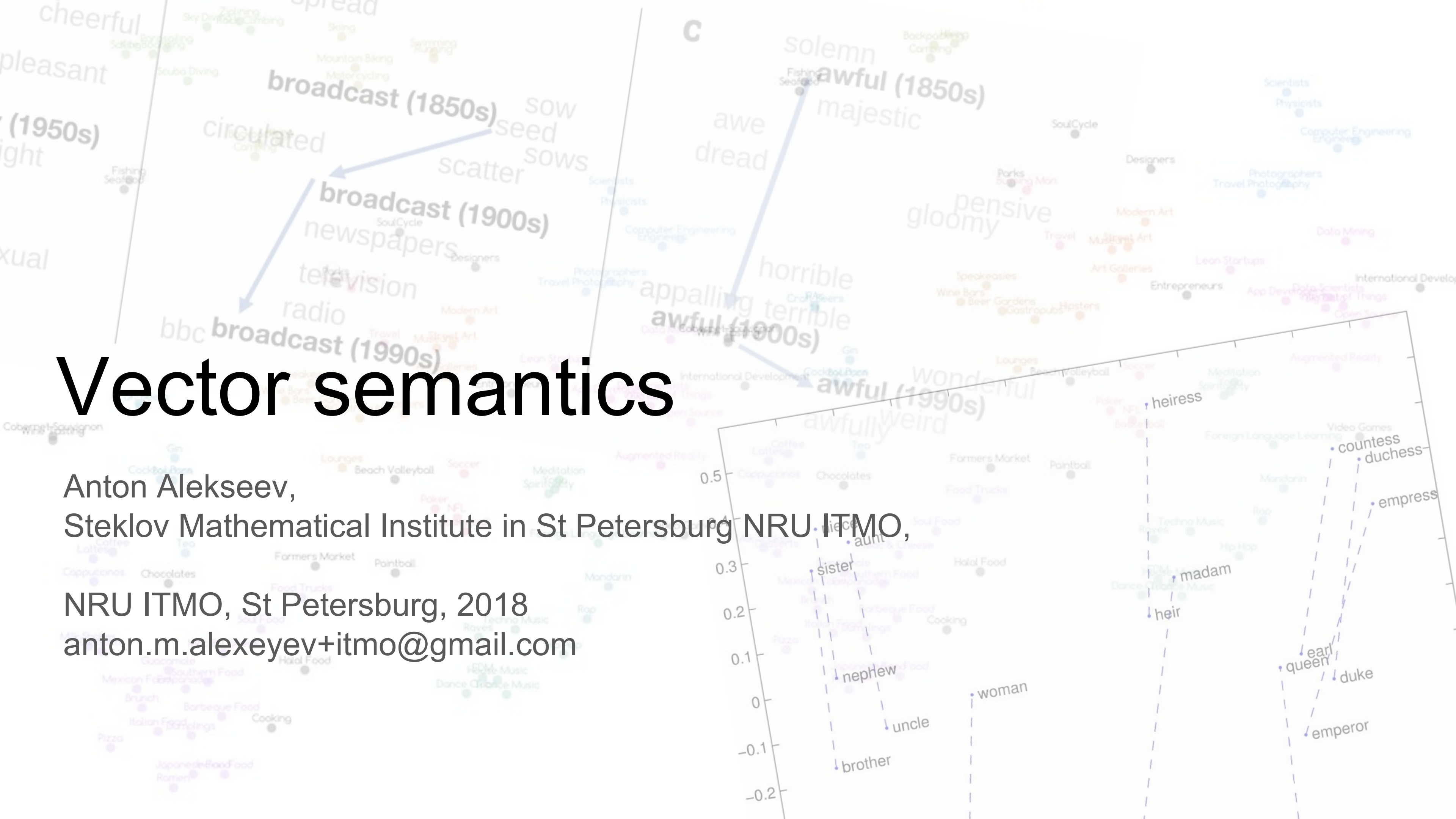
$$I(X;Y) = D_{\mathrm{KL}}(p(x,y) \| p(x)p(y)).$$



Markov models, information theory and why we care about it all - II

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg NRU ITMO,

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



Vector semantics

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg, NRU ITMO,

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

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

Words in similar contexts *have similar meaning*

Nothing of **things** that have been **said** was **important**.
Nothing of **stuff** that has been **announced** was **useful**.

I bought X in the nearest shop.
I came home, hung X on the balcony and hung my trousers on it.
The prisoners used X to escape from their cell’s window.

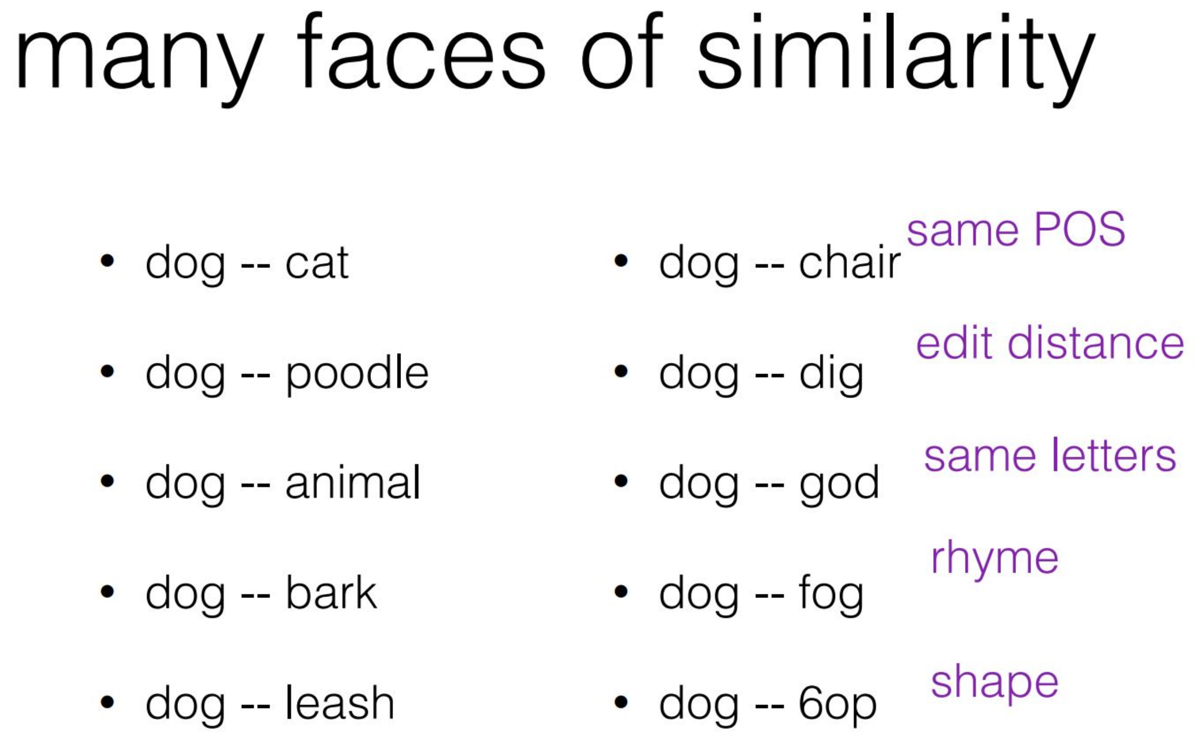
Can you guess **what is X**? Any ideas of the properties it has?

What is ‘similarity’?

- **first-order co-occurrence**
  (syntagmatic association)
  Words close in the text, such as:
  ‘drank’ -- and ‘lemonade’/’water’/’tea’

- **second-order co-occurrence**
  (paradigmatic association)
  Words having similar neighbours:
  ‘Tatra’ and ‘Carpathian’, ‘to pet’ and ‘to stroke’

What IS ‘similarity’?



many faces of similarity

• dog -- cat	• dog -- chair	same POS
• dog -- poodle	• dog -- dig	edit distance
• dog -- animal	• dog -- god	same letters
• dog -- bark	• dog -- fog	rhyme
• dog -- leash	• dog -- 6op	shape

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

Every word needs a ‘meaning’ vector

What for?

1. **Most important**: something like transfer learning: instead of BoW
   (this way we reuse information from another (possibly bigger) text collection
   [and it actually helps])

2. A tool for finding synonyms and other ‘related’ words in some sense

3. Language research tool!
   a. Example: semantic evolution for historians:
      https://nlp.stanford.edu/projects/histwords/
      (there are a few earlier works BTW)

4. **Fun!** quizzes (odd one out), rewriting Great Russian Novels, etc.

Ideas:
how do we learn to find words with similar meaning?

We’ve met before: term-document matrix

But now we care about rows, not columns
(word vectors, not document vectors)

	Zemfira -- Nebomoreoblaka	**Sky -- Wikipedia**	**Fabrika -- The Sea Calls**	**Eugene Onegin Chapter 1**	**Anastasia -- The Queen of Gold Sand**
sky	6	60		2	
sea	6		10	4	1
cloud	6	18			
love				6	
sand			1		2

We’ve met before: term-document matrix

But now we care about rows, not columns
(word vectors, not document vectors)

	Zemfira -- Nebomoreoblaka	**Sky -- Wikipedia**	**Fabrika -- The Sea Calls**	**Eugene Onegin Chapter 1**	**Anastasia -- The Queen of Gold Sand**
sky	6	60		2	
sea	6		10	4	1
cloud	6	18			
love				6	
sand			1		2

We’v

But nov

(word v

```
>>> import numpy as np
>>> sea = np.array([6,0,10,4,1])
>>> sand  =np.array([0,0,1,0,2])
>>> cloud = np.array([6,18,0,0,0])

>>> cosine = lambda x,y: x.dot(y) / np.linalg.norm(x) / np.linalg.norm(y)

>>> cosine(sea, sand) > cosine(sea, cloud)
True
>>> cosine(sea, sand) > cosine(sand, cloud)
True
```

sky	6	60		2	
sea	6		10	4	1
cloud	6	18			
love				6	
sand			1		2

Discussion: term-document matrix

- We need A LOT of representative documents, otherwise the approach won’t work

- Dimensionality depends on the text collection size

- Distribution of topics should not be ‘skewed’

- To solve this, maybe we could split documents into subdocuments...
  E.g. sentences? (**NO!** why?)

Discussion: term-document matrix

- We need A LOT of representative documents, otherwise the approach won’t work
- Dimensionality depends on the text collection size
- Distribution of topics should not be ‘skewed’
- To solve this, maybe we could split documents into subdocuments...
  E.g. sentences? (**NO!** why?)

However, looking at smaller **CONTEXT** may be a great idea

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

Way better: word-word (word-context) matrix

We count how many times the word occured in the same context with other words (*e.g. in a [-2, 2] window*)

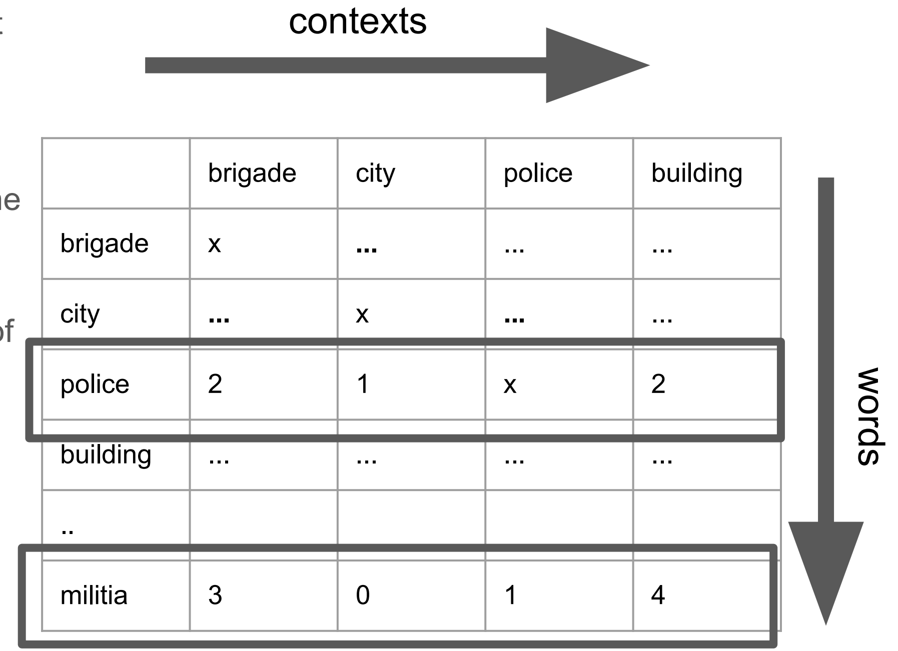
...in Admiralteysky district, St Petersburg. A 37-year old citizen was arrested by [a **police** brigade at around] midnight close to the station of…

...an explosion rambled on Tuesday night close to the entrance of [the **police** station in the] city of Helsingborg...

...the unknown with cold steel arms attacked [the police brigade at the] gas station...

In [Vyborg, police station might eventually] catch fire...

We get sparse vectors with a large number of dimensions


contexts
words

	brigade	city	police	building
brigade	x	**...**	...	...
city	**...**	x	**...**	...
police	2	1	x	2
building	...	...	...	...
..				
militia	3	0	1	4

Similar words have almost the same row cells filled

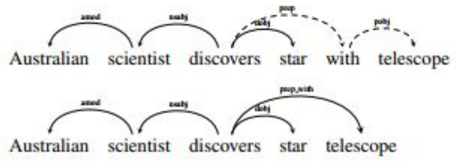
Way better: word-word (word-context) matrix

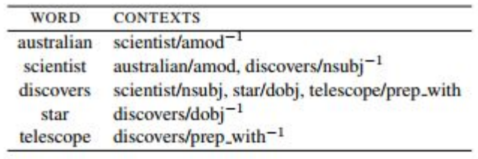
Important: there are many ways to **define ‘co-occurence’**

E.g., one can choose a ‘syntactically motivated’ part of a sentence as a context -- instead of a window

see. "**Dependency-Based Word Embeddings**", Omer Levy and Yoav Goldberg, 2014

(however, this paper is on dense vectors, the ones we haven’t yet discussed)


amod
nsubj
prep
dobj
pobj
Australian scientist discovers star with telescope
amod
nsubj
prep_with
dobj
Australian scientist discovers star telescope



WORD	CONTEXTS
australian	scientist/amod^{-1}
scientist	australian/amod, discovers/nsubj^{-1}
discovers	scientist/nsubj, star/dobj, telescope/prep_with
star	discovers/dobj^{-1}
telescope	discovers/prep_with^{-1}

The choice of context window defines vector’s properties

1. Small window -- ~ ‘syntactic’ similarity
2. Larger window -- ~ ‘meaning’ similarity

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

Example: HAL (Hyperspace Analogue to Language)

Oldschool example: ‘window approach’ where we increment counters for ALL pairs of words in a window

This way the words that are closer to each other in the window get more ‘weight’

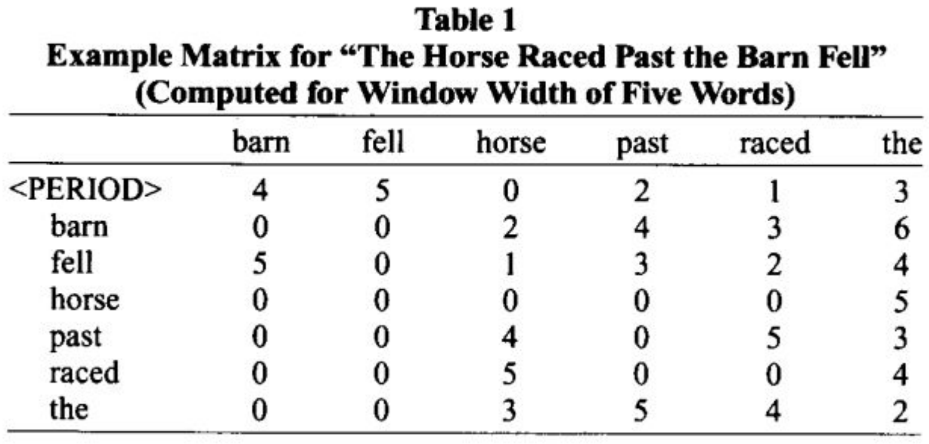


Table 1
Example Matrix for “The Horse Raced Past the Barn Fell” (Computed for Window Width of Five Words)

	barn	fell	horse	past	raced	the
<PERIOD>	4	5	0	2	1	3
barn	0	0	2	4	3	6
fell	5	0	1	3	2	4
horse	0	0	0	0	0	5
past	0	0	4	0	5	3
raced	0	0	5	0	0	4
the	0	0	3	5	4	2

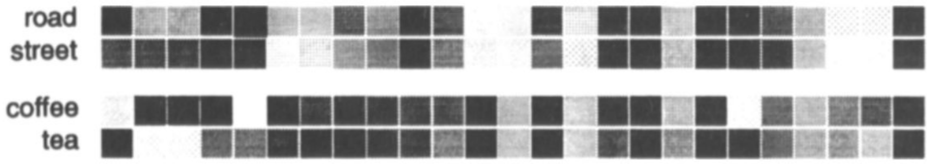


Figure 1. Gray-scaled 25-element co-occurrence vectors.

Lund, K., Burgess, C. & Atchley, R. A. (1995). Semantic and associative priming in a high-dimensional semantic space. Cognitive Science Proceedings (LEA), 660-665.

Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. Behavior Research Methods, Instrumentation, and Computers, 28, 203-208.

Example: HAL (Hyperspace Analogue to Language)

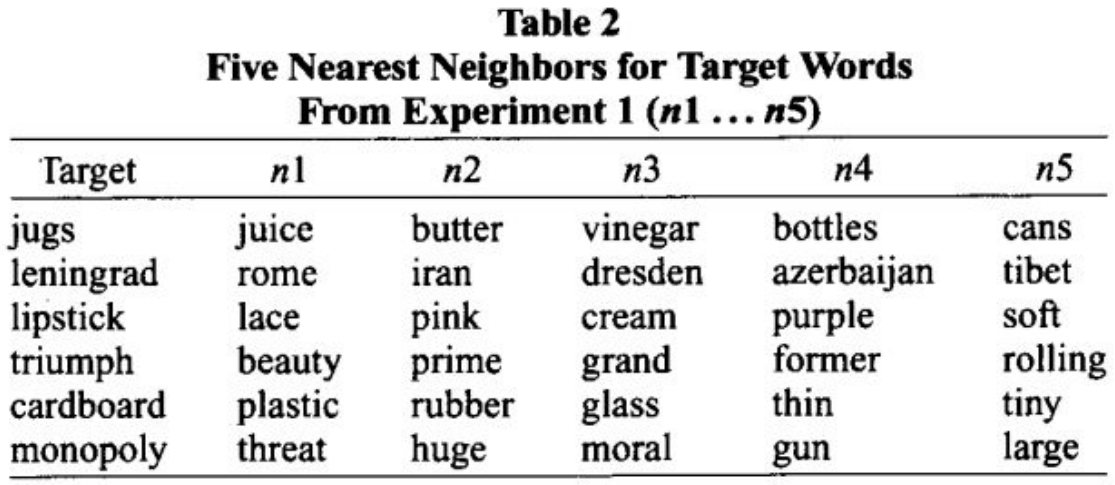


Table 2
Five Nearest Neighbors for Target Words From Experiment 1 (*n*1 . . . *n*5)

Target	*n*1	*n*2	*n*3	*n*4	*n*5
jugs	juice	butter	vinegar	bottles	cans
leningrad	rome	iran	dresden	azerbaijan	tibet
lipstick	lace	pink	cream	purple	soft
triumph	beauty	prime	grand	former	rolling
cardboard	plastic	rubber	glass	thin	tiny
monopoly	threat	huge	moral	gun	large

Lund, K., Burgess, C. & Atchley, R. A. (1995). Semantic and associative priming in a high-dimensional semantic space. Cognitive Science Proceedings (LEA), 660-665.

Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. Behavior Research Methods, Instrumentation, and Computers, 28, 203-208.

Disadvantages of ‘simple counts’

Counts assign large values to ‘useless’ words (in terms of meaning) such as prepositions, articles, etc. However they do not add any useful information.

Question: any ideas on how to modify **weight(word, context),** so that useless yet frequent words won’t have large weight?

Let’s use ‘importances’ as weights

We know at least two ways to do it

For term-document vectors (discussed earlier):

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

...simple **idf** is also valid.

For term-term case:

$$\mathrm{pmi}(x; y) \equiv \log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}.$$

Also: be careful when removing stop-words!

PMI-weighted word-context matrix

Estimating probabilities as frequencies of occurrences within the same window for a given word

contexts →

words ↓

	brigade	city	police	building
brigade	x	...	...	...
city	...	x	...	...
police	2	1	x	2
building	...	...	...	...
..				
militia	3	0	1	4

$$\mathrm{PMI}(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)}$$

p(w) = count(police, *) / all =
sum of ‘police’ row / sum of matrix elements

p(c) = count(*, station) / all =
sum of ‘station’ row / sum of matrix elements

p(w, c) = count(police station) / all =
2 / sum of matrix elements

Positive PMI (PPMI)

We often have to deal with rare words (e.g. one in a million), thus checking whether two events with probabilities lower than 10^-6 (estimated as a simple fraction of counts) is a bad idea :(

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \quad p_{i*} = \frac{\sum_{j=1}^{C} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \quad p_{*j} = \frac{\sum_{i=1}^{W} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

$$\text{PPMI}_{ij} = \max(\log_2 \frac{p_{ij}}{p_{i*}p_{*j}}, 0)$$

Problem: (P)PMI “likes” rare events

Omer Levy, Yoav Goldberg, Ido Dagan introduced a trick to deal with it in 2015:

$$\mathrm{PPMI}_\alpha(w,c) = \max\left(\log_2 \frac{P(w,c)}{P(w)P_\alpha(c)}, 0\right)$$

$$P_\alpha(c) = \frac{count(c)^\alpha}{\sum_c count(c)^\alpha}$$

...Inspired by similar ideas in word2vec and GloVe implementations

A value of 0.75 showed the best performance on all tasks

(though may need tuning on your task!)

Levy, O., Goldberg, Y., and Dagan, I. (2015). Improving distributional similarity with lessons learned from word embeddings. TACL, 3, 211–225.

Other weighting schemes

Student’s t-test: estimation how far from each other are observed mean and expected mean

“Can we reject this hypothesis?”

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}} \longrightarrow$$

$$P(a,b) = P(a)P(b)$$

$$\text{t-test}(a,b) = \frac{P(a,b) - P(a)P(b)}{\sqrt{P(a)P(b)}}$$

One can use this statistic for collocations extraction as well

Почему так можно?
Manning, C. D. and Schutze, H. (1999). ¨ Foundations of Statistical Natural Language Processing. MIT Press.
Curran, J. R. (2003). From Distributional to Semantic Similarity. PhD thesis

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

Vector closeness estimation

We already know one way to do it

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}}$$

Another view on this:

1. scalar product is a ‘weighted set intersection cardinality’
2. we need the denominator as a way to heal scalar product’s tendency to grow because of the large vector values (possibly few)

Vector closeness estimation - 2

“Soft” Jaccard distance (context = set element)

$$\mathrm{sim}_{\mathrm{Jaccard}}(\vec{v}, \vec{w}) = \frac{\sum_{i=1}^{N} \min(v_i, w_i)}{\sum_{i=1}^{N} \max(v_i, w_i)}$$

Normalize vectors so that the sum of values of each equals to 1 and compute the KL-divergence between them

$$D(P||Q) = \sum_{x} P(x) \log \frac{P(x)}{Q(x)}$$

Is that OK?

Vector closeness estimation - 2

“Soft” Jaccard distance (context = set element)

$$\mathrm{sim}_{\mathrm{Jaccard}}(\vec{v},\vec{w}) = \frac{\sum_{i=1}^{N} \min(v_i, w_i)}{\sum_{i=1}^{N} \max(v_i, w_i)}$$

Normalize vectors so that the sum of values of each equals to 1 and compute the KL-divergence between them

$$D(P||Q) \;=\; \sum_{x} P(x) \log \frac{P(x)}{Q(x)}$$

We may have zeros we can’t divide by **or take logarithm of**

Vector closeness estimation* - 3

Symmetric distance based on Kullback-Leibler divergence:

$$D(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

Jensen-Shannon divergence, a sum of KL-d between each distribution and an average distribution

$$JS(P||Q) = D(P|\frac{P+Q}{2}) + D(Q|\frac{P+Q}{2})$$

In our case it looks like this

$$\mathrm{sim}_{\mathrm{JS}}(\vec{v}||\vec{w}) = D(\vec{v}|\frac{\vec{v}+\vec{w}}{2}) + D(\vec{w}|\frac{\vec{v}+\vec{w}}{2})$$

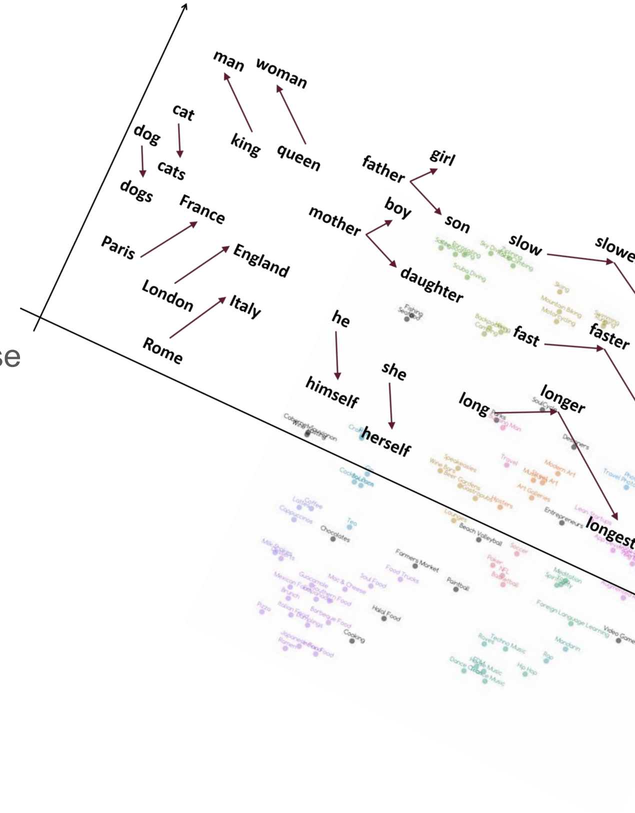
Lecture plan

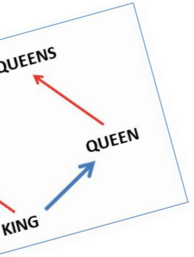
1. ~~Sparse vectors~~
   a. ~~“Term-document” approach~~
   b. ~~“Term-term” approach~~
      i. ~~Construction~~
      ii. ~~HAL~~
   c. ~~Weighting~~
   d. ~~Semantic similarity estimation~~
   e. Quality evaluation
2. Dense vectors
   a. Matrix decomposition
   b. “Predictive” approaches

Word vectors quality evaluation

1. **Extrinsic evaluation**
   the best way to estimate word vectors quality for practical tasks. E.g.:
   a. short texts classification
   b. any other useful task :)

2. **Intrinsic evaluation**
   a. mainstream: evaluation on pairs of words that are 'similar' in some sense
   b. mainstream: syntactic/semantic analogy tasks
   c. clustering words labeled with 'groups' (+computing purity)
   d. ...a few more ideas

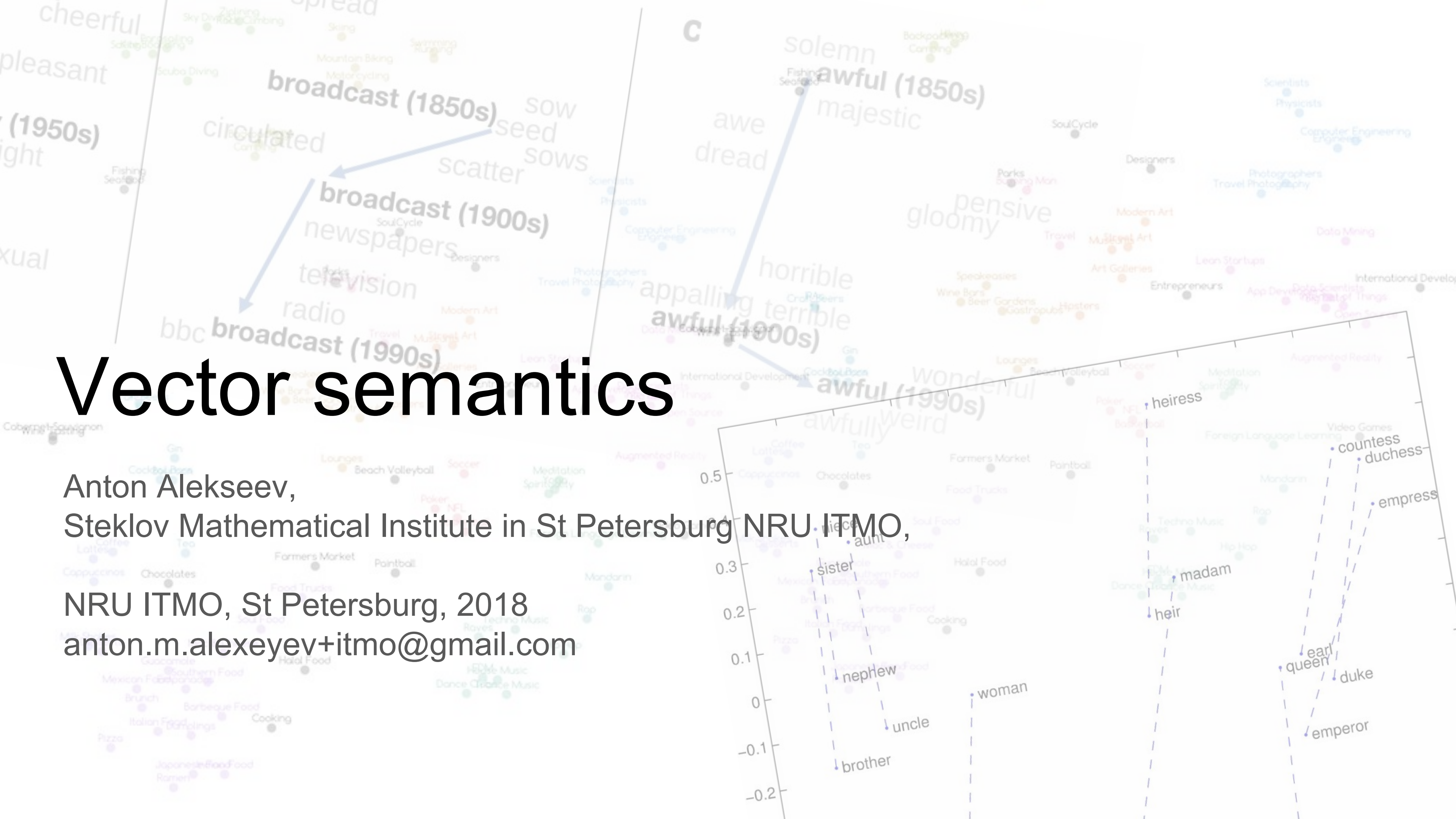




See also: Schnabel, Tobias & Labutov, Igor & Mimno, David & Joachims, Thorsten. (2015). Evaluation methods for unsupervised word embeddings. 298-307. 10.18653/v1/D15-1036.
Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors M Baroni, G Dinu, G Kruszewski Proceedings of Association for Computational Linguistics (ACL) 1

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



Vector semantics

Anton Alekseev,
Steklov Mathematical Institute in St Petersburg NRU ITMO,

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

to be continued...

On entropy of sequences
and its connection with perplexity

please see Martin/Jarfsky ed.3 **4.7**
https://web.stanford.edu/~jurafsky/slp3/4.pdf

additional slides on that are in Russian

Энтропия последовательности

- Часто нам важен текст как последовательность
- Нет проблем: для всякого языка L, задающего последовательности длины n

$$H(w_1, ..., w_n) = - \sum_{(w_1,...,w_n) \in L} p(w_1, ..., w_n) log_2 p(w_1, ..., w_n)$$

- Энтропия языка с последовательностями бесконечной длины

$$H(L) = \lim_{n \to \infty} \frac{1}{n} H(w_1, ... w_n) =$$

$$= - \lim_{n \to \infty} \frac{1}{n} \sum_{(w_1,...,w_n) \in L} p(w_1, ..., w_n) log_2 p(w_1, ..., w_n)$$

Энтропия последовательности

- Часто нам важен текст как последовательность
- Нет проблем: для всякого языка L, задающего последовательности длины n

$$H(w_1, ..., w_n) = - \sum_{(w_1, ..., w_n) \in L} p(w_1, ..., w_n) log_2 p(w_1, ..., w_n)$$

- Энтропия языка с последовательностями бесконечной длины

$$H(L) = \lim_{n \to \infty} \frac{1}{n} H(w_1, ... w_n) =$$

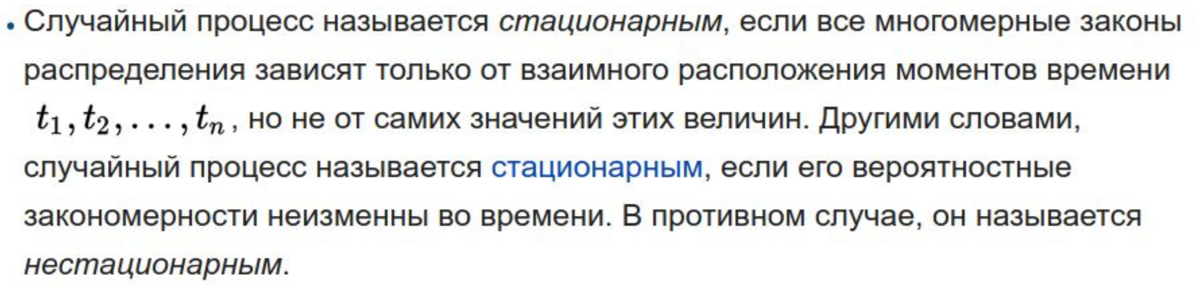
$$= - \lim_{n \to \infty} \frac{1}{n} \sum_{(w_1, ..., w_n) \in L} p(w_1, ..., w_n) log_2 p(w_1, ..., w_n)$$

УЖАС, как это считать?!

Стационарность стохастического процесса

Стохастический процесс называется **стационарным**, если вероятности последовательностей инвариантны относительно сдвигов позиций во времени

Википедия

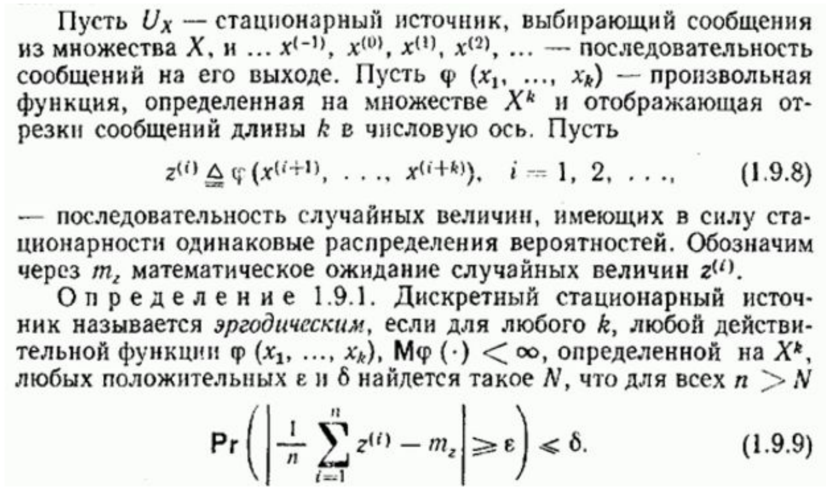


- Случайный процесс называется *стационарным*, если все многомерные законы распределения зависят только от взаимного расположения моментов времени $t_1, t_2, \ldots, t_n$, но не от самих значений этих величин. Другими словами, случайный процесс называется стационарным, если его вероятностные закономерности неизменны во времени. В противном случае, он называется *нестационарным*.

Для естественного языка это, очевидно, не так, но иногда в рамках моделей мы можем себе позволить такое приближение

Эргодический стационарный стохастический процесс

В. Д. Колесник, Г. Ш. Полтырев
“Курс теории информации”



Пусть U_X — стационарный источник, выбирающий сообщения из множества X, и $\dots x^{(-1)}, x^{(0)}, x^{(1)}, x^{(2)}, \dots$ — последовательность сообщений на его выходе. Пусть $\varphi(x_1, \dots, x_k)$ — произвольная функция, определенная на множестве X^k и отображающая отрезки сообщений длины k в числовую ось. Пусть

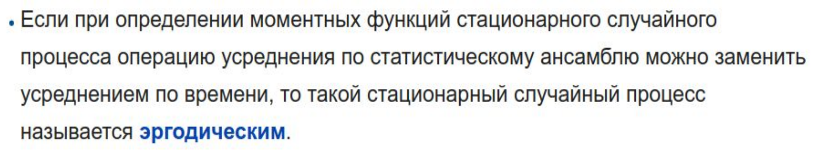
$$z^{(i)} \triangleq \varphi(x^{(i+1)}, \dots, x^{(i+k)}), \quad i = 1, 2, \dots, \tag{1.9.8}$$

— последовательность случайных величин, имеющих в силу стационарности одинаковые распределения вероятностей. Обозначим через m_z математическое ожидание случайных величин $z^{(i)}$.

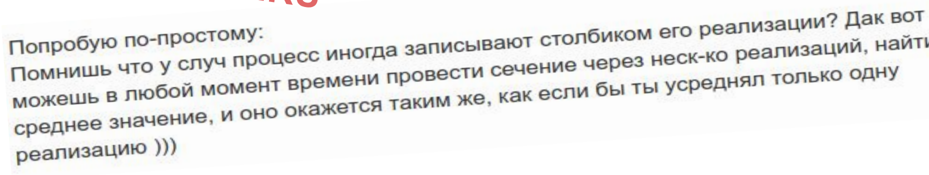
Определение 1.9.1. Дискретный стационарный источник называется *эргодическим*, если для любого k, любой действительной функции $\varphi(x_1, \dots, x_k)$, $\mathsf{M}\varphi(\cdot) < \infty$, определенной на X^k, любых положительных ε и δ найдется такое N, что для всех $n > N$

$$\Pr\left(\left|\frac{1}{n}\sum_{i=1}^{n} z^{(i)} - m_z\right| \geq \varepsilon\right) < \delta. \tag{1.9.9}$$

Википедия



- Если при определении моментных функций стационарного случайного процесса операцию усреднения по статистическому ансамблю можно заменить усреднением по времени, то такой стационарный случайный процесс называется **эргодическим**.



Ответы Mail.RU

Попробую по-простому:
Помнишь что у случ процесс иногда записывают столбиком его реализации? Дак вот
можешь в любой момент времени провести сечение через неск-ко реализаций, найт
среднее значение, и оно окажется таким же, как если бы ты усреднял только одну
реализацию)))

Энтропия последовательности

- **Теорема Шэннона-МакМиллана-Бреймана** спешит на помощь: при стационарности и эргодичности последовательности верно, что

$$H(L) = -\lim_{n\to\infty} \frac{1}{n} log_2 p(w_1, ...w_n)$$

...то есть мы можем *просто* взять достаточно длинную последовательность для хорошей оценки

- То же верно при таких же допущениях и для перекрёстной энтропи

$$H(p,q) = -\lim_{n\to\infty} \frac{1}{n} log_2 q(w_1, ...w_n)$$

Зачем всё это: энтропия и перплексия

- Вспомним

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x_1, ..., x_{i-1})$$

- Выпишем формулу перплексии

$$PP(W) = \sqrt[n]{\frac{1}{p(x_1, ..., x_n)}} = 2^{-\frac{1}{n} log_2 p(x_1,...,x_n)} =$$

$$= 2^{-\frac{1}{n}\sum_{i=1}^{n} \log P(x_i | x_1 ... x_{i-1})} \rightarrow 2^{H(W)}$$

- Перплексия — экспонента кросс-энтропии языка, которую мы оцениваем на достаточно длинном тексте