

Introduction into Natural Language Processing

aka DeepLearningless NLP (almost)

Lectures: Anton Alekseev, Steklov Mathematical Institute in St Petersburg
NRU ITMO, St Petersburg, 2018

Logistics of the course

‘Who are you to _____ lecture me?’

Lectures

Anton Alekseev, researcher at Steklov Mathematical Institute
ex-Yandex, ex-SofitLabs (chatbots), ex-nativeroll.tv (ML in video ads)

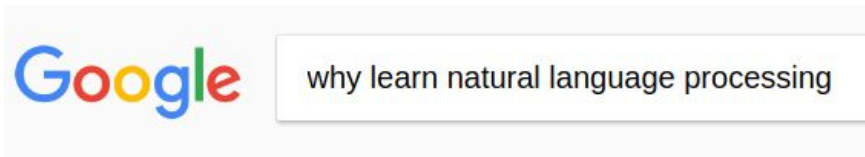
Labs and tests

Ksenia Buraya, researcher at NRU ITMO

Questions regarding lectures and the course in general:

anton.m.alexeyev+itmo@gmail.com

Why know anything about NLPProc?



Google doesn't even...
(could be so due to personalization though)

- NLPProc is fun
- We communicate with intelligent machines, they communicate with us
- Web Data ~ Collective Intelligence
- NLP engineers are paid, like, money
- etc

How to learn Natural Language Processing - Quora
<https://www.quora.com/How-do-I-learn-Natural-Language-P...> Перевести эту страницу
How do I learn **Natural Language Processing** ... The best way to learn about it is to go do it. Rob's suggestions of an intro course or a tutorial is a great way to get ...

What is the best way to learn NLP? - Quora
<https://www.quora.com/what-is-the-best-way-to-learn-NLP> - Перевести эту страницу
For **NLP** in-depth understanding of both algorithms for processing linguistic information and the underlying computational properties of natural languages is needed.



Course features

Fast-paced introductory course on NLProcessing

- practice-oriented
- statistical approaches are always preferred
- mostly based on lecturer's hands-on experience
- memes



An Incomplete List of NLP Processing Tasks

Language modeling
Part-of-speech tagging
Named entity recognition
Text (topic) classification
Keyword extraction
Spelling checking
Syntax parsing
Dependency parsing
Machine translation
Stemming
Lemmatization
Distributional semantics
Text generation
Text clustering

Wikification
QA systems, dialogue systems
Plagiarism detection
Morpheme analysis
Grammar check
Hyphenation
Relation extraction
Entity linking
Sentiment analysis
Topic modeling
Text summarization
Semantic role labeling
Information retrieval
Speech Recognition / Synthesis

Stuff NOT covered in the course

1. Almost NO neural networks
 - a. **Deep Learning for NLP** requires one more course
 - [A Primer on Neural Network Models for Natural Language Processing](#), Yoav Goldberg
 - [Natural Language Processing with Deep Learning](#), Christopher Manning
 - b. as of early 2k18, there is arguably no such task in NLP where ANNs are a huge breakthrough in terms of quality
(I mean, the way it was with AlexNet for vision)
2. Oldschool classic theories and methods from early CompLing
3. *With this simple trick you can write your own chatbot in just 2 weeks*
4. A few important yet complex/esoteric tasks
(e.g., text summarization, coreference resolution, QA-systems, ...)
 - a. one can't fit everything into NLP 101;
 - b. these are problems the average engineer doesn't solve every day.

Used/Recommended materials

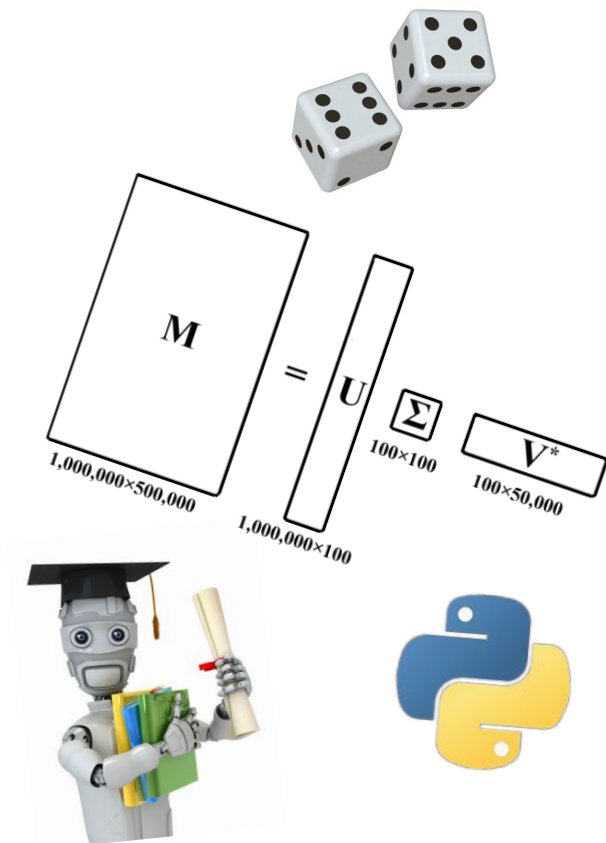
1. **Introduction to Information Retrieval**,
Chr. Manning, Pr. Raghavan and H. Schütze. Cambridge University Press. 2008.
2. **Foundations of Statistical Natural Language Processing**,
Chr. Manning, H. Schütze, MIT Press. Cambridge, MA: May 1999.
3. **Speech and Language Processing**
Daniel Jurafsky, James H. Martin
4. **Прикладная и КОМПЬЮТЕРНАЯ ЛИНГВИСТИКА**. Николаев И.С., Митренина О.В., Ландо Т.М.
(ред.). Изд.2 Прикладная и компьютерная лингвистика URSS. 2017. 320 с.
5. [LxMLS 2015](#) tutorials, lectures, code
6. HPLabs @ CSCenter NLP Course (Al. Ulanov), Autumn 2013
7. Certain papers and reviews + personal experience
8. *Wikipedia* (pics, formulae, pseudocode :)

Other useful sources

1. Coursera (Chr. Manning - D.Jurafsky, M. Collins, D. Radev), Stepik (П. Браславский)
2. <http://ods.ai/> - Russian-speaking data scientists community
3. <http://nlpub.org/> - NLProc wiki in Russian

Prerequisites

1. Probability theory basics
2. Linear algebra basics
3. Machine learning basics
(or a huge interest towards it)
4. The skill to hand in homeworks on time
5. Python basics



Scores

Cumulative:

1. Tests
2. Labs (homeworks)
3. Final exam

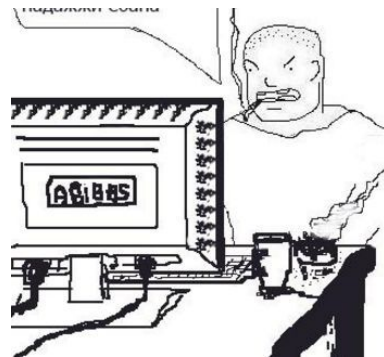
Any questions?

Introduction into NLP

things i **have to** tell you, because other teachers usually do
minimum minimorum

Natural language

- CompLing vs NLP
- Linguistics studies language
 - Phonetics
 - Morphology
 - Syntax
 - Semantics
 - Pragmatics (arguable division)



- What's so hard?

aaaaa how do i represent words and characters as numbers, there are too many words, it is hard to obtain a representative sample, polysemy! word order, all sorts and flavors of ambiguities, no word for some senses in some

Roots, bloody roots

- Can't skip this: Alexander Markov Sr., 1913, "Eugene Onegin", experimenting: Markov chains for consonants/vowels distribution
- Established opinion: CompLing was born in 1950s thanks to the machine translation task (both in the USA and the USSR)



Georgetown experiment

60+ sentences, 6 rules

1954, IBM + Georgetown University, "the task will be solved in 3-5 years" :)

Vyelyichyina ugla opryedyelayetsya otnoshyenyiyem dlyini dugi k radiusu.

Magnitude of angle is determined by the relation of length of arc to radius.

Myezhdunarodnoye ponyimaniye yavlyayetsya vazhnim faktorom v ryeshyenyiyi polyityichyeskix voprosov.

International understanding constitutes an important factor in decision of political questions.

https://en.wikipedia.org/wiki/Georgetown%E2%80%93IBM_experiment

http://rvb.ru/soft/articles/eonegin_1913.htm



Must know

- **ELIZA**: virtual assistant, rogerian psychotherapist parody, based on keywords extraction, pattern matching and slot filling
MIT AI lab, Joseph Weizenbaum, 1966

See wiki: https://en.wikipedia.org/wiki/ELIZA_effect

young woman: You are not very aggressive but I think you don't want me to notice that.

eliza: What makes you think *I am not very aggressive*?

young woman: You don't argue with me.

eliza: Why do you think *I don't argue with you*?

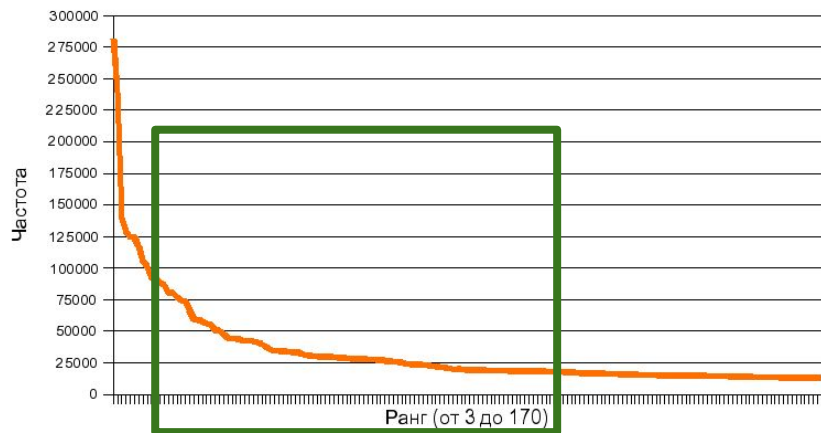
young woman: You are afraid of me.

eliza: Does it please you to believe *I am afraid of you*?

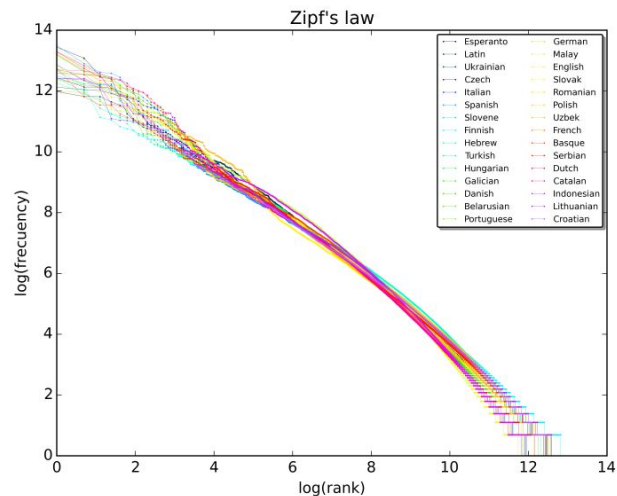
<https://web.stanford.edu/group/SHR/4-2/text/dialogues.html>

Zipf law [1949]: rank * freq ~ const

(Jean-Baptiste Estoup (1868–1950), Felix Auerbach (1856–1933), George Kingsley Zipf (1902–1950))



Russian Wikipedia words frequencies, sorted (starting with the third one)



First 10 million words in 30 national wikipedia texts (Oct. 2015) (log-log scale!).

Must know: progress

50-e: first attempts, Information Theory, Formal Grammars

60-70-e: 'Syntactic Structures', AI, bayesian models, first corpora

80-e: structured models (speech!), first distributional semantics approaches, data-driven research

90-e: models evaluation tracks, applications for wide range of users

2000-e: web! data! machine learning, unsupervised approaches

2010-e: Deep Learning feast, tons of applications in various fields

Normal



Weird



Creep



Irrimediabile



Introduction into Information Retrieval

Lectures: Anton Alekseev, Steklov Mathematical Institute in St Petersburg
NRU ITMO, St Petersburg, 2018

Why do we even discuss this?

Information retrieval (even **text retrieval**) is not a part of NLP, however

- intersection is large,
- classical IR tricks are widely used in NLP,
- a good and hopefully inspiring practical start for the NLP course,
- other lecturers do this $\bar{_}(_)(_)_/\bar{_}$ (yeah, cargo cult!)



ad-hoc-retrieval - text is not the only option

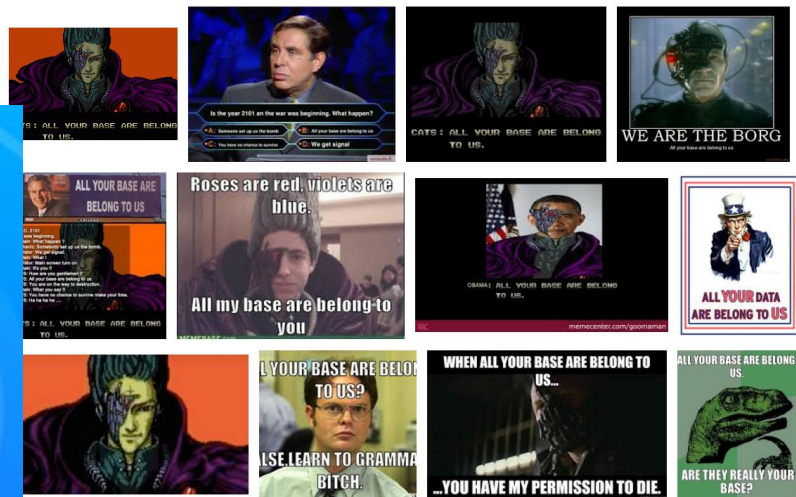
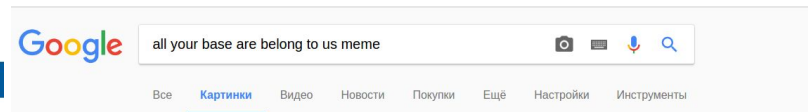
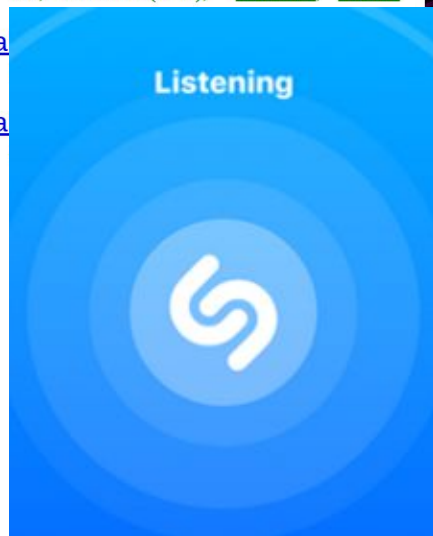


list ontologies matching ontology search

http://wiki.creativecommons.org/Special:ExportRDF/Case_Studies/Radiohead
SemanticWebDocument, RDFXML, 2013-02-09, 7K, ontoRatio(0.43), [metadata](#), [cached](#)

http://wiki.creativecommons.org/Special:ExportRDF/Case_Studies/Radiohead
SemanticWebDocument, RDFXML, 2013-02-09,

http://wiki.creativecommons.org/Special:ExportRDF/Case_Studies/Radiohead
SemanticWebDocument, RDFXML, 2013-02-10,



...but this course is focused on text processing

ad-hoc search task clumsy definition

D - a set of documents (that is, texts + possibly some metadata)

T - a set of terms (words)

Q - a set of queries, also a sequence of terms

We believe there exists a function **Rel**: $\mathbf{Q} \times \mathbf{D} \rightarrow \mathbf{R}$ that can provide all pairs of queries and documents with an estimate of relevance (a measure to what extent the user's information need (represented as a query) is satisfied by the document)

The goal is to find best-matching documents (in terms of **Rel**). On the fly.

Cunning plan

1. “Download the Internet”
2. Invent Rel
3. Scan the database for every query and compute Rel for every query-document pair
4. ????????
5. PROFIT!

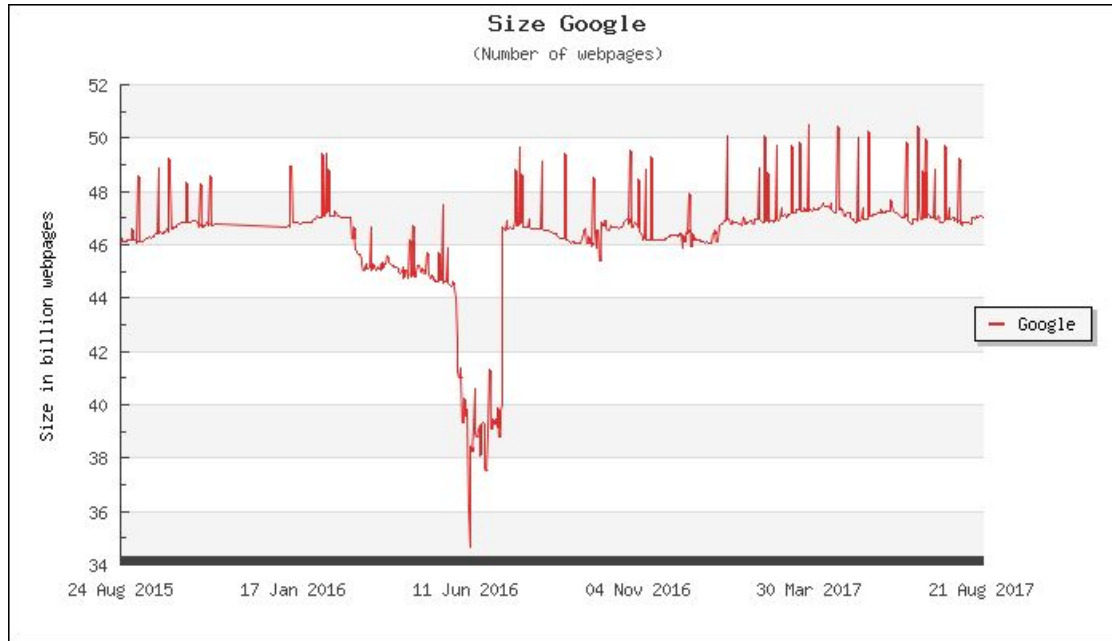


GENIUS!

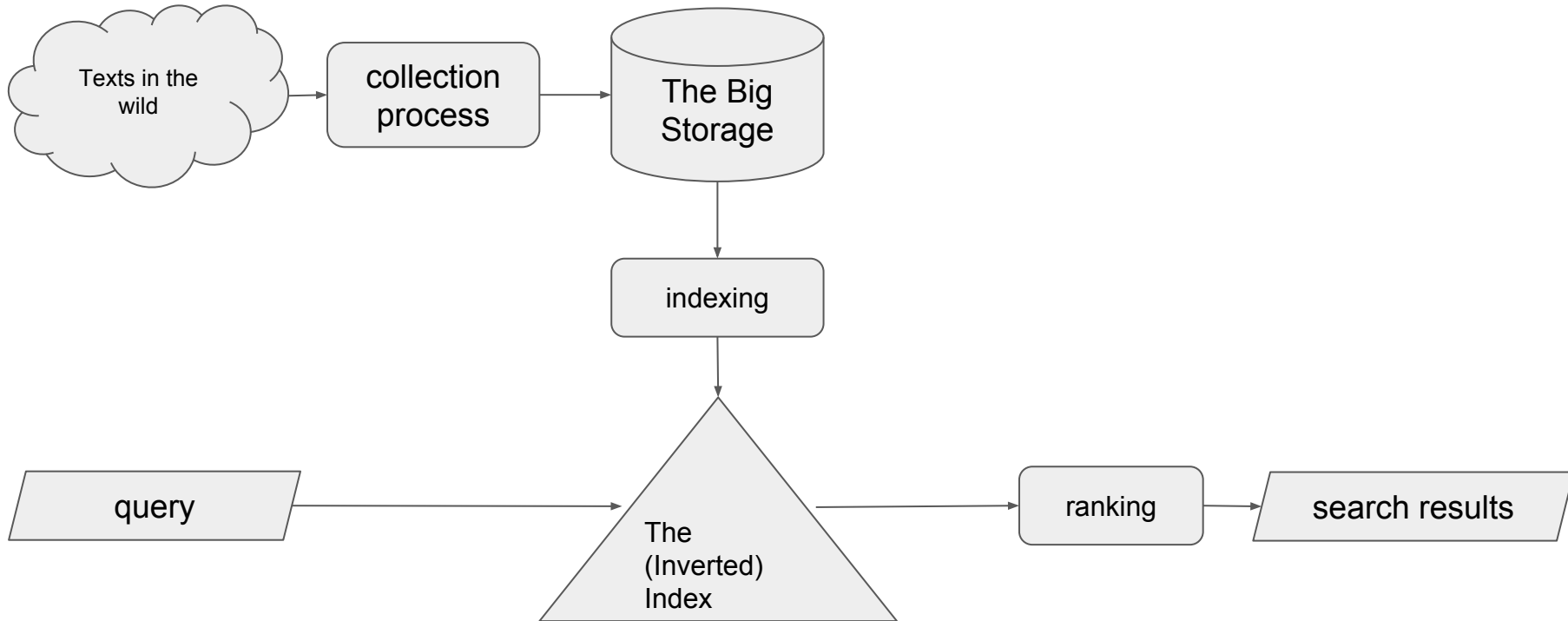


Well, no.

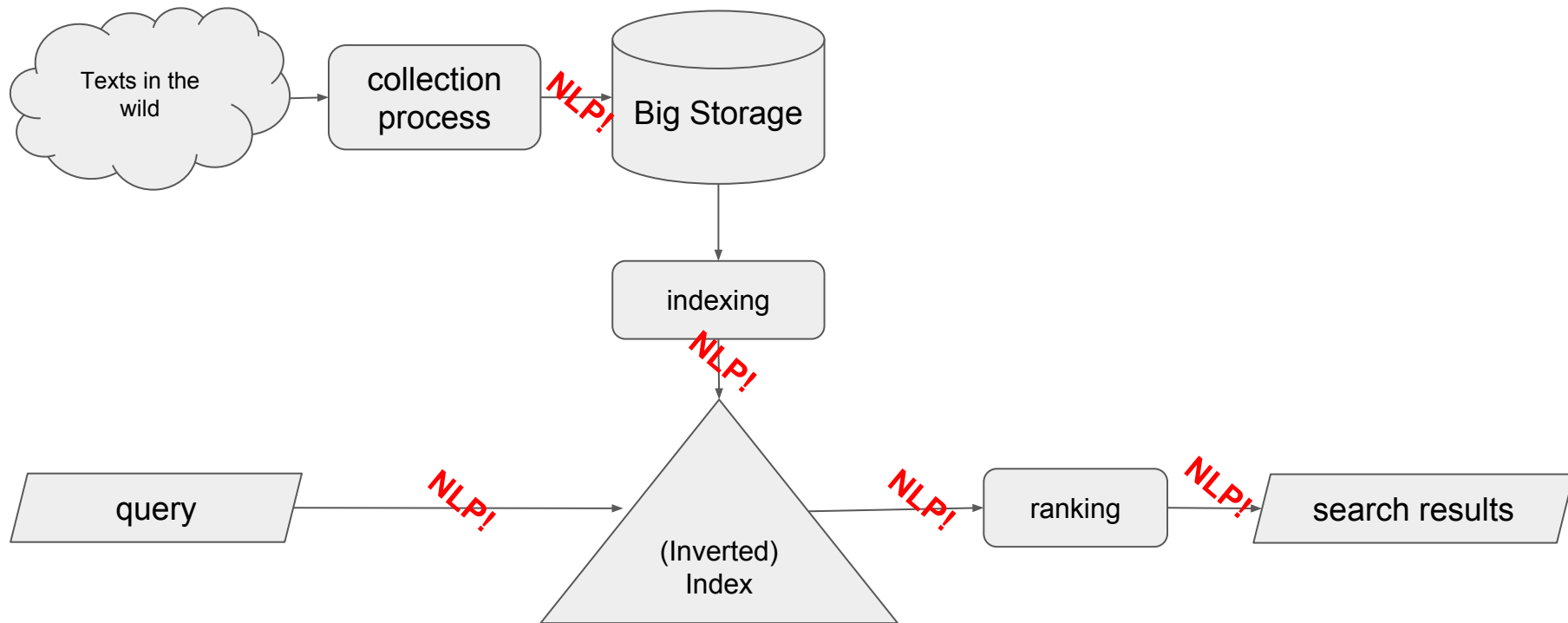
(you should be terrified: Google would have to perform a full scan for 45+ billion records per query!)



How it is usually done



How it is usually done



Information retrieval stages

1. Collecting and cleaning data
2. Preparing documents
3. Document indexing
4. Query processing
5. Documents retrieval by query
6. Document ranking
7. Search results preparation

Also: (8) personalization, (9) query suggestions, ...

Documents preparation

Initially we have 'dirty' texts

«На берегу пустынных волн»

на берегу пустынных волн

«Russian language!»

на берег пустынный волна

(или: **на** берег пустын волн)

берег пустынный волна

lemmatization (base forms)

stemming (stripping suffixes, etc.)

not even a word

Data preparation: stemming

Morphing words so that all possible forms of the word would turn into a single item, stem. Can be solved as a language-independent task.

- Usually when we say ‘stemming’ we mean cutting words (removing suffixes, prefixes, etc.) so that only the **common part of all forms of the target word remains**



Classics: Porter stemmer

The first paper on stemming

Lovins, Julie Beth (1968). Development of a Stemming Algorithm. Mechanical Translation and Computational Linguistics

More widely spread and well-known is this one

C.J. van Rijsbergen, S.E. Robertson and M.F. Porter, 1980. New models in probabilistic information retrieval. London: British Library. (British Library Research and Development Report, no. 5587).

M.F. Porter, 1980, An algorithm for suffix stripping, Program, 14(3) pp 130–137.

Позже разработал фреймворк для разработки стеммеров -- Snowball.

The *rules* for removing a suffix will be given in the form

(condition) S1 -> S2

The 'condition' part may also contain the following:

- *S - the stem ends with S (and similarly for the other letters).
- *v* - the stem contains a vowel.
- *d - the stem ends with a double consonant (e.g. -TT, -SS).
- *o - the stem ends cvc, where the second c is not W, X or Y (e.g. -WIL, -HOP).

IZ	-> IZE	siz(ed)	-> size
(*d and not (*L or *S or *Z))	-> single letter	hopp(ing)	-> hop

Porter stemmer usage example

```
>>> from nltk.stem.porter import
>>> stemmer = PorterStemmer()
>>> words = "hello,dear,students,!,who,cares,about,linguistics,?".split(",")
>>> words =
"hello,dear,students,!,who,cares,about,linguistics,?,let's,evaluate,all,the,corpora,!".split
(",")

>>> " ".join([stemmer.stem(w) for w in words])
"hello dear student ! who care about linguist ? let' evalu all the corpora !"

>>> stemmer.stem("corpuses")
'corpus'
```

Preparing documents: lemmatization

- conversion to infinitive forms of words

Usually: dictionary-based approach + morphological tricks!

“World languages”: **nlTK**, **spacy**, **pattern**, ...

Russian: **mystem 3.0** / **pymorphy2**

Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.

Korobov M.: Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts, pp 320-332 (2015).

I.Segalovich, Yandex. A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine

Yandex MyStem Demo

```
./mystem.exe -e utf-8 -i --format json -d --weight
```

```
варкалось. хливкие шорьки
```

```
[  
  {"text":"варкалось","analysis":[  
    {  
      "Wt":0.6584257483,  
      "Lex":"варкаться",  
      "gr":"V,несов,нп=прош,ед,изъяв,сред",  
      "Qual":"bastard"  
    }  
  ]},  
  {"text":"хливкие","analysis":[  
    {"wt":0.9958436489,"lex":"хливкий","gr":"A=вин,мн,полн,неод","qual":"bastard"},  
    {"wt":0.9958436489,"lex":"хливкий","gr":"A=им,мн,полн","qual":"bastard"}  
  ]},  
  {"text":"шорьки","analysis":[  
    {"wt":0.3092010319,"lex":"шорька","gr":"S,жен,неод=вин,мн","qual":"bastard"},  
    {"wt":0.3092010319,"lex":"шорька","gr":"S,жен,неод=род,ед","qual":"bastard"},  
    {"wt":0.3092010319,"lex":"шорька","gr":"S,жен,неод=им,мн","qual":"bastard"}  
  ]  
]
```



Yandex MyStem Demo: morph. ambiguity

\$./local/bin/mystem


мама мыла раму

мама{мама}мыла{мыло|мыть}раму{рама|рам}

NOUN (soap)



VERB (wash) past tense



\$./local/bin/mystem -d


мама мыла раму

мама{мама}мыла{мыть}раму{рама}

Enabling disambiguation



VERB (wash) past tense



Information retrieval stages

1. Collecting and cleaning data
2. Preparing documents
3. Document indexing
4. Query processing
5. Documents retrieval by query
6. Document ranking
7. Search results preparation

Also: (8) personalization, (9) query suggestions, ...

Indexing (for boolean search)

So one does not simply scan the web

The minimal entity in search are **terms** -- so let us first learn how to retrieve documents containing certain terms and term combinations, that is, “boolean retrieval”

Information need = “would love to read wiki page on manul or maine coon”

Boolean query = “wiki” AND (“maine coon” OR “manul”)



Indexing (for boolean search)

“wiki” AND (“manul” OR “maine coon”)

Long bit vectors + bitmasking for search!

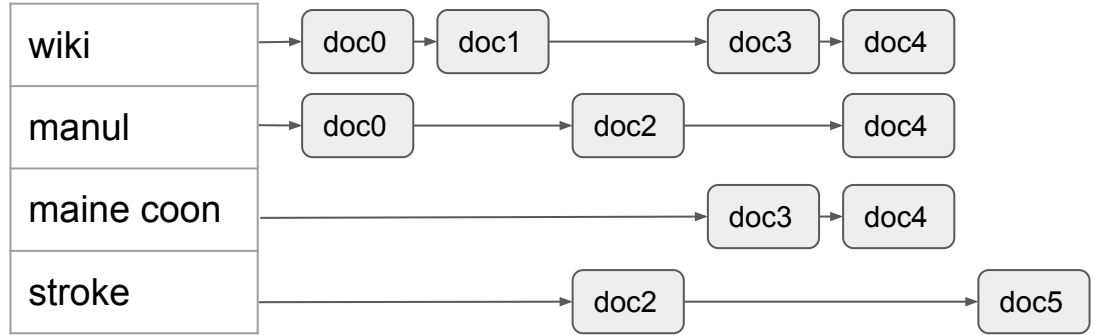
Is this an acceptable solution?

	wiki	manul	maine coon	stroke
doc0	1	1	0	0
doc1	1	0	0	0
doc2	0	1	0	1
doc3	1	0	1	0
doc4	1	1	1	1

Indexing (for boolean search)

“wiki” AND (“maine coon” OR “manul”)

	wiki	manul	m.coon	stroke
doc0	1	1	0	0
doc1	1	0	0	0
doc2	0	1	0	1
doc3	1	0	1	0
doc4	1	1	1	1
doc5	0	0	0	1



A dictionary, with linked lists of document metainformation records (position in text, true word form, frequency of the term in concern in the document etc.) **ordered by document IDs**

How would you solve this retrieval problem algorithmically?

Inverted indices

AND: intersection of sorted lists

OR: union of sorted lists

Dictionary can be stored in memory, lists can be stored on disk

Multiple natural tricks for performance and quality boost:

- store lists compressed and unpack on the fly when reading
- store IDs diffs, not IDs themselves
- add forward-links once in a while (skip-lists)
- store positions of terms in the doc
(allows to use distances between terms from query)

Information retrieval stages

1. Collecting and cleaning data
2. Preparing documents
3. Document indexing
4. Query processing
5. Documents retrieval by query
6. Document ranking
7. Search results preparation

Also: (8) personalization, (9) query suggestions, ...

Query preprocessing

Preparing the query the way we did with the document -- and we have the boolean retrieval system all set and ready

Also a good idea in practice

- add extra similar and relevant terms (**query expansion**)
- classification of query intention to understand which index to use (there may be many specific ones)

However, there are many more tricks, which are out of scope of this course



GOOD
ENOUGH!

Information retrieval stages

1. Collecting and cleaning data
2. Preparing documents
3. Document indexing
4. Query processing
5. Documents retrieval by query
6. Document ranking
7. Search results preparation

Also: (8) personalization, (9) query suggestions, ...

Ranking is the brain of the search engine

Академия Яндекса

**Ranking is hard!
(Yandex employee)**

Ранжирование — это сложно



2:28:42 / 3:25:10

Why is that hard?

jaguar|

jaguar напиток

jaguar

jaguar xf

📍 **Jaguar** в Санкт-Петербурге - отзывы, фото, телефоны,...

maps.yandex.ru > jaguar

Jaguar в Санкт-Петербурге - отзывы, фото, телефоны, адреса с рейтингом, отзывами и фотографиями. Адреса, телефоны, часы работы, схема проезда.

🖼️ **jaguar** — смотрите картинки

yandex.ru/images > jaguar ▾



📺 **Jaguar** — поддержанные и новые авто в Санкт-Петербурге

[Запчасти](#) [Объявления](#) [Отзывы](#) [Каталог](#) [Дилеры](#)

auto.ru > Jaguar

Большая база объявлений о продаже автомобилей **Jaguar**. Полная информация об автомобилях — фотографии, отзывы, характеристики и цены.

an animal, a vehicle, a drink



Why so hard?

One have to solve multiple problems simultaneously

1. Matching document and query
2. Document quality
3. Matching user's interests and behavioral patterns*
4. Search results diversification
(one of the “Jaguar case” solutions)
5. ...

Ranking: the task

The goal is to sort search results so that the most relevant would be at the top of the list

Can be treated as a task of learning the relevance function

Rel: $Q \times D \rightarrow R$

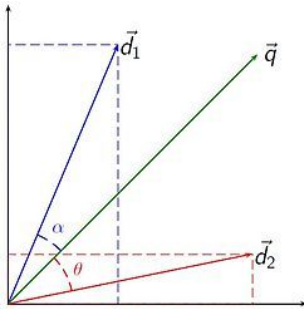


Classics: Vector Space Model (VSM)

Every text and every query is represented as a vector of the same fixed number of dimensions, then documents vector representations are sorted by the distance/closeness to the query vector

$$d_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$$

$$q = (w_{1,q}, w_{2,q}, \dots, w_{n,q})$$



Example:

$$\cos \theta = \frac{\mathbf{d}_2 \cdot \mathbf{q}}{\|\mathbf{d}_2\| \|\mathbf{q}\|}$$

$$\text{sim}(d_j, q) = \frac{\mathbf{d}_j \cdot \mathbf{q}}{\|\mathbf{d}_j\| \|\mathbf{q}\|} = \frac{\sum_{i=1}^N w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^N w_{i,j}^2} \sqrt{\sum_{i=1}^N w_{i,q}^2}}$$

How to build vectors: counters

Query / document:

“На берегу пустынных волн...”

абакан	абырвалг	азкабан	аксакал	аксолоть	аксон	берег	волна	заноза	...
0	0	0	0	0	0	1	1	0	...

(tip: never ever forget to try this way of representing texts, may work surprisingly well, sometimes way better than all those trendy things everyone loves)

How to build vectors: tf-idf

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

Is the word met frequently in the document?
(more is “better”)

Are there many docs with this word in the collection?
(the smaller the number of documents the better)

$$\text{tf}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$\text{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Variants of term frequency (TF) weight

weighting scheme	TF weight
binary	0, 1
raw count	$f_{t,d}$
term frequency	$f_{t,d} / \sum_{t' \in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Variants of inverse document frequency (IDF) weight

weighting scheme	IDF weight ($n_t = \{d \in D : t \in d\} $)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(1 + \frac{N}{n_t} \right)$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t} \right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

Explaining it with 'theory'

TF-IDF(q, d) — мера релевантности документа d запросу q

n_{dw} (term frequency) — число вхождений слова w в текст d ;

N_w (document frequency) — число документов, содержащих w ;

N — число документов в коллекции D ;

N_w/N — оценка вероятности встретить слово w в документе;

$(N_w/N)^{n_{dw}}$ — оценка вероятности встретить его n_{dw} раз;

$P(q, d) = \prod_{w \in q} (N_w/N)^{n_{dw}}$ — оценка вероятности встретить

в документе d слова запроса $q = \{w_1, \dots, w_k\}$ *чисто случайно*;

Оценка релевантности запроса q документу d :

$$-\log P(q, d) = \sum_{w \in q} \underbrace{n_{dw}}_{\text{TF}(w, d)} \underbrace{\log(N/N_w)}_{\text{IDF}(w)} \rightarrow \max.$$

TF(w, d) = n_{dw} — term frequency;

IDF(w) = $\log(N/N_w)$ — inverted document frequency.

Probability to see the term w in a document

Probability to see the term w in a document the number of times it actually shows up in a document (n_{dw})

Probability to see query q terms in a document **at random**

Relevance: the larger, **the less the 'randomness' of query terms** occurrence in the document

BM is for Best Match

The famous **Okapi BM25** — a family of IDF-like functions that were popular in early web search engines

Q — query

D — document

avgdl — average document length

The rest — tuneable parameters

$$\text{score}(D, Q) = \sum_{i=1}^n \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgdl}})},$$

Machine learning, of course

Learning to Rank!

Rel(d,q): D x Q => R

- **Elementwise** approach:

We have relevance scores defined in train set; fitting to them

- **Pairwise** approach

$\text{Rel}(d, q) < \text{Rel}(d', q)$, fitting the function given pairs

- **Listwise** approach

Lists of documents $\{\mathbf{d}_i\}$, sorted by relevance for \mathbf{q}

Information retrieval stages

1. Collecting and cleaning data
2. Preparing documents
3. Document indexing
4. Query processing
5. Documents retrieval by query
6. Document ranking
7. Search results preparation

Also: (8) personalization, (9) query suggestions, ...

Search results preparation

Search results representation for the user is terribly important

Snippets preparation is a separate NLP task, similar to automatic text summarization problem

Web Images Video News More ▾ Anytime ▾

Humpty Dumpty - Wikipedia

en.wikipedia.org/wiki/Humpty_Dumpty ▾

Humpty Dumpty sat on a wall, **Humpty Dumpty** had a big fall. All the king's horses and all the king's men Couldn't put **Humpty** together again. ...

Humpty Dumpty Sat On A Wall - Pandora Hearts Wiki

pandorahearts.wikia.com/wiki/Humpty_Dumpty_Sat_On_A_Wall ▾

Humpty Dumpty Sat On A Wall is the 53rd chapter of Jun Mochizuki's Pandora Hearts. The chapter...

Humpty Dumpty (House) - Wikipedia

[en.wikipedia.org/wiki/Humpty_Dumpty_\(House\)](https://en.wikipedia.org/wiki/Humpty_Dumpty_(House)) ▾

"**Humpty Dumpty**" is the third episode of the second season of House, which premiered on Fox on September 27, 2005. Dr. Lisa Cuddy's longtime handyman Alfredo falls off ...

Humpty Dumpty | Muppet Wiki | FANDOM powered by Wikia

muppet.wikia.com/wiki/Humpty_Dumpty ▾

"**Humpty Dumpty sat** on a wall, **Humpty Dumpty** had a great fall. All the king's horses and all the king's men couldn't put **Humpty** together again. ...

SEARCH

Web Images Videos News More ▾ Tools

SafeSearch ▾

About 64,500,000 results

Web Results

[All your base are belong to us - Wikipedia](#)

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

"All your base are belong to us" is a popular Internet meme based on a broken English ("English") phrase found in the opening cutscene of the 1992 Mega Drive port of the 1989 arcade video game Zero Wing.

[All Your Base Are Belong to Us | Know Your Meme](#)

knowyourmeme.com/memes/all-your-base-are-belong-to-us

"All Your Base Are Belong to Us" is a classic catchphrase that has been spawning derivatives at

Standard datasets

- **The Cranfield collection**
(kinda ancient, 1400 texts on aeronautics, judgements on relevance 0/1)
- **TREC**
(text retrieval conference; various tasks and datasets; new challenges every year)
- **CLEF**
(cross language evaluation forum; tasks for various languages)

How to build your own IR engine

Lots of open source software providing IR-out-of-box.

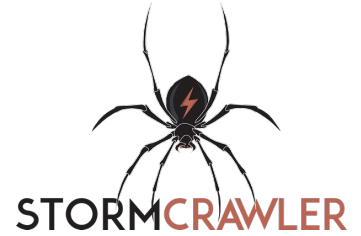
Python

- Scrapy, requests/urllib + BeautifulSoup
- PyLucene, Xapian

Java

- Crawling: Nutch, StormCrawler, ...
- Retrieval: **Lucene** (Java library), Elasticsearch (incl. Lucene), Solr

Software for doing IR



Open Source Search Server



...the list is not even close to being a complete one

Conferences and schools

- International **ACM SIGIR**
Conference on Research and Development in Information Retrieval
- **ECIR**
- WWW (and the like)
- Schools: ESSIR, RuSSIR

- Many more:
<http://www.wikicfp.com/cfp/call?conference=information%20retrieval>

Other tasks in web search

- Query correction/augmentation
- Query expansion: morphology, semantics
- Search/suggestions personalization
- Dealing with queries semantic ambiguity
- Fact extraction
- Smart suggestions
- Queries classification
- Document clustering
- Duplicate detection
- Virus / spam detection
- Events detection
- ...

Final remarks on IR's impact on NLP

- It is mainly thanks to IR community's efforts that evaluation of data processing algorithms became a common practice (esp. in NLP)
- IR \neq NLP, but IR + NLP = <3
interconnections & common tricks
- Web search is probably the most successful case of applying NLP in production (tthough the number grows)

Overview

1. Text search nuts and bolts
2. Stemming and lemmatization
3. Boolean retrieval
4. Vector Space Model
- 5. Cosine similarity**
- 6. TF-IDF**
7. A few popular ranking quality evaluation metrics

Introduction into Information Retrieval

Anton Alekseev
anton.m.alexeyev+itmo@gmail.com

Thanks for useful comments go to: Denis Kiryanov

Extra: ranking quality evaluation

$$DCG(\text{ranking for query } q) = \sum_{j=1}^{N_q} \frac{rel_j}{\log_2 j + 1}$$

$$nDCG(\dots) = \frac{DCG(\text{ranking for query } q)}{DCG(\text{ideal ranking for query } q)}$$

$$MAP = \frac{\sum_{q=1}^Q AveP(q)}{Q}$$

$$pfound = \sum_{i=1}^n pLook[i] * pRel[i]$$

$$pLook[i] = pLook[i-1] * (1 - pRel[i-1]) * (1 - pBreak)$$

<http://romip.ru/russir2009/slides/yandex/lecture.pdf>

http://romip.ru/romip2009/15_yandex.pdf

Discounted cumulative gain (DCG)

(penalty for highly relevant results being put to the bottom of the ranked list)

nDCG (normalized ...)

(normalizing DCG (perfect ranking in the denominator); known to correlate well with human judgements)

Precision@K (-> MAP@k)

(count of 'true relevant' items found in predicted TOP-k; MAP - averaging for all ranked lists)

pFound

modeling the user viewing search results page from top to bottom, and sometimes leaving it; **pRel** - relevance probability estimate