Statistical Natural Language Processing

P. Rychlý

NLP Centre, FI MU, Brno

September 21, 2021

Statistical Natural Language Processing

- statistics provides a summary (of a text)
- highlights important or interesting facts
- can be used to model data
- foundation of estimating probabilities
- fundamental statistics: size (+ domain, range)

Statistical Natural Language Processing

- statistics provides a summary (of a text)
- highlights important or interesting facts
- can be used to model data
- foundation of estimating probabilities
- fundamental statistics: size (+ domain, range)

	lines	words	bytes
Book 1	3,715	37,703	223,415
Book 2	1,601	16,859	91,031

Word list

- list of all words from a text
- list of most frequent words
- words, lemmas, senses, tags, domains, years ...

Book 1	Book 2
the, and, of, to, you, his, in, said, that, I, will, him, your, he, a, my, was, with, s, for, me, He, is, , , it, them, be, The, all, , have, from, , on, her, , , , are, their, were, they, which, , t, up, , had, there	the, I, to, a, of, is, that, , you, he, and, said, was, , in, it, not, me, my, have, And, are, one, for, But, his, be, The, It, at, all, with, on, will, as, very, had, this, him, He, from, they, , so, them, no, You, do, would, like

Word list

- list of all words from a text
- list of most frequent words
- words, lemmas, senses, tags, domains, years ...

Book 1	**Book 2**
the, and, of, to, you, his, in, said, that, I, will, him, your, he, a, my, was, with, s, for, me, He, is, **father**, , it, them, be, The, all, **land**, have, from, , on, her, , **son**, , are, their, were, they, which, **sons**, t, up, , had, there	the, I, to, a, of, is, that, **little**, you, he, and, said, was, , in, it, not, me, my, have, And, are, one, for, But, his, be, The, It, at, all, with, on, will, as, very, had, this, him, He, from, they, **planet**, so, them, no, You, do, would, like

Word list

- list of all words from a text
- list of most frequent words
- words, lemmas, senses, tags, domains, years ...

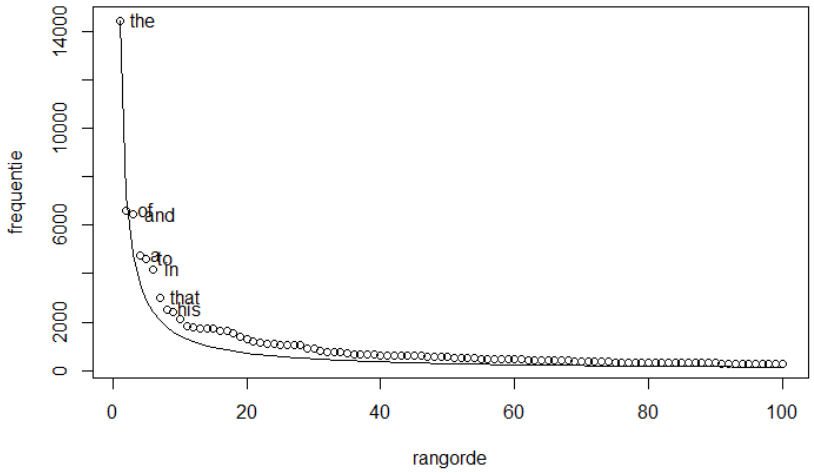
Book 1	**Book 2**
the, and, of, to, you, his, in, said, that, I, will, him, your, he, a, my, was, with, s, for, me, He, is, **father**, **God**, it, them, be, The, all, **land**, have, from, **Jacob**, on, her, **Yahweh**, **son**, **Joseph**, are, their, were, they, which, **sons**, t, up, **Abraham**, had, there	the, I, to, a, of, is, that, **little**, you, he, and, said, was, **prince**, in, it, not, me, my, have, And, are, one, for, But, his, be, The, It, at, all, with, on, will, as, very, had, this, him, He, from, they, **planet**, so, them, no, You, do, would, like

Frequency

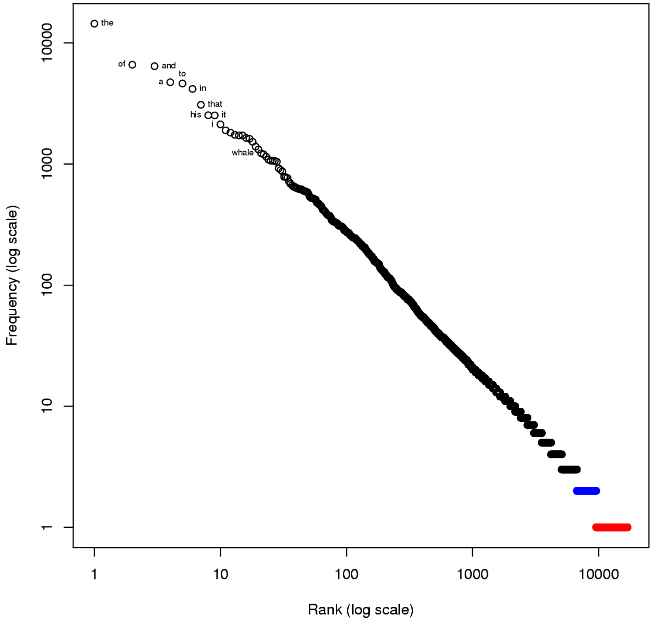
- number of occurrences (raw frequency)
- relative frequency (hits per million)
- document frequency (number of documents with a hit)
- reduced frequency (ARF, ALDf)
  $1 < \textit{reduced} < \textit{raw}$
- normalization for comparison
- hapax legomena ($= 1$ hit)

Zipf's Law

- rank-frequency plot
- rank × frequency = constant



Zipf's Law



the
of
and
to
a
in
that
his
it
i
whale
Frequency (log scale)
10000
1000
100
10
1
Rank (log scale)
1
10
100
1000
10000

Keywords

- select only *important* words from a word list
- compare to reference text (norm)
- simple math score:

$$score = \frac{freq_{focus} + N}{freq_{reference} + N}$$

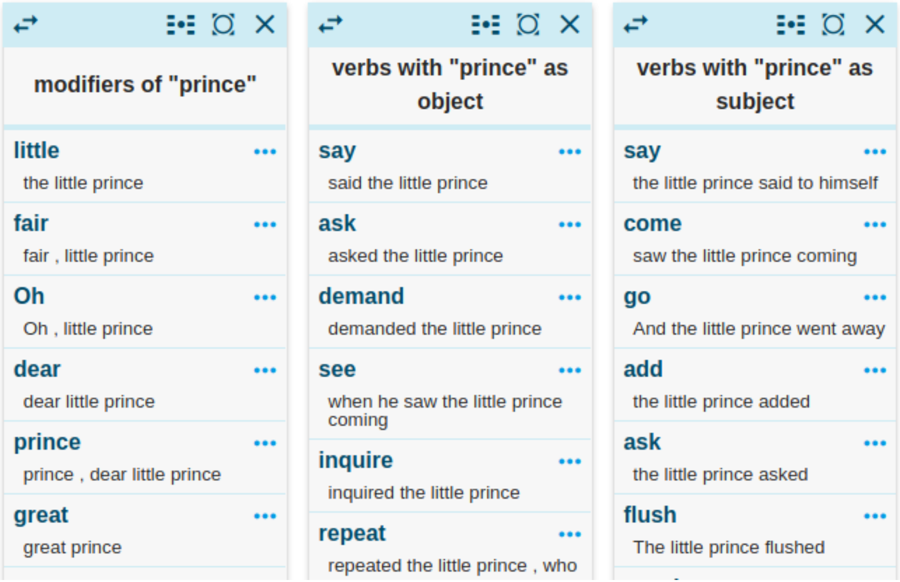
Genesis	**Little Prince**
son God father Jacob Yahweh Joseph Abraham wife behold daughter	prince planet flower little fox never too drawing reply star

Collocations

- meaning of words is defined by the context
- collocations a *salient* words in the context
- usually not the most frequent
- filtering by part of speech, grammatical relation
- compare to *reference* = context for other words
- many statistics (usually single use only) based on frequencies
- MI-score, t-score, χ^2, ...
- logDice – scalable

$$logDice = 14 + log \frac{f_{AB}}{f_A + f_B}$$

Collocations of Prince



modifiers of "prince"

- **little** — the little prince
- **fair** — fair , little prince
- **Oh** — Oh , little prince
- **dear** — dear little prince
- **prince** — prince , dear little prince
- **great** — great prince

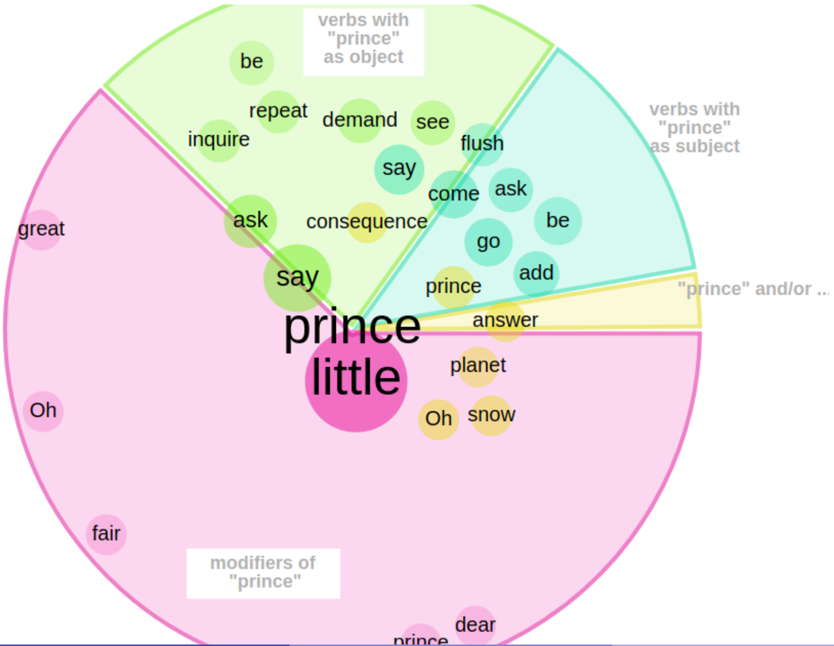
verbs with "prince" as object

- **say** — said the little prince
- **ask** — asked the little prince
- **demand** — demanded the little prince
- **see** — when he saw the little prince coming
- **inquire** — inquired the little prince
- **repeat** — repeated the little prince , who

verbs with "prince" as subject

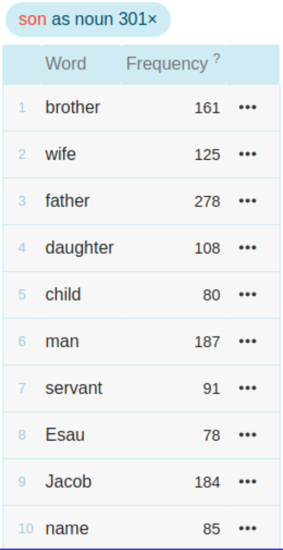
- **say** — the little prince said to himself
- **come** — saw the little prince coming
- **go** — And the little prince went away
- **add** — the little prince added
- **ask** — the little prince asked
- **flush** — The little prince flushed

Collocations of Prince



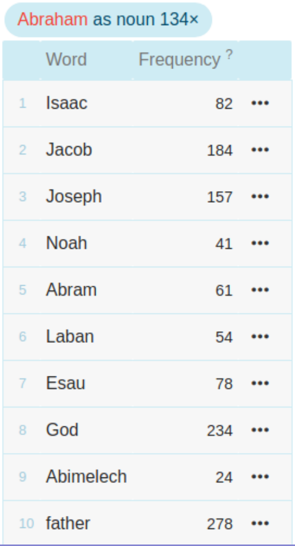
Thesaurus

- comparing collocation distributions
- counting same context



son as noun 301×

	Word	Frequency ?	
1	brother	161	•••
2	wife	125	•••
3	father	278	•••
4	daughter	108	•••
5	child	80	•••
6	man	187	•••
7	servant	91	•••
8	Esau	78	•••
9	Jacob	184	•••
10	name	85	•••



Abraham as noun 134×

	Word	Frequency ?	
1	Isaac	82	•••
2	Jacob	184	•••
3	Joseph	157	•••
4	Noah	41	•••
5	Abram	61	•••
6	Laban	54	•••
7	Esau	78	•••
8	God	234	•••
9	Abimelech	24	•••
10	father	278	•••

Multi-word units

- meaning of some words is completely different in the context of specific co-occurring word
- *black hole*, is not black and is not a hole
- strong collocations
- uses same statistics with different threshold
- better to compare context distribution instead of only numbers
- terminology – compare to a reference corpus

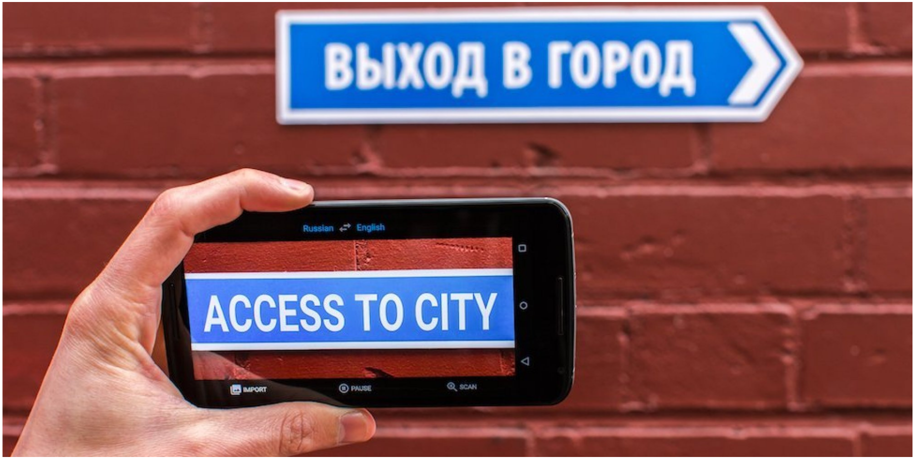
Language models—what are they good for?

- assigning scores to sequences of words
- predicting words
- generating text

$\Rightarrow$

- statistical machine translation
- automatic speech recognition
- optical character recognition

OCR + MT


ВЫХОД В ГОРОД
Russian ⇄ English
ACCESS TO CITY
IMPORT
PAUSE
SCAN

Language models – probability of a sentence

- LM is a probability distribution over all possible word sequences.
- What is the probability of utterance of s?

Probability of sentence

p_{LM}(Catalonia President urges protests)
p_{LM}(President Catalonia urges protests)
p_{LM}(urges Catalonia protests President)
…

Ideally, the probability should strongly correlate with fluency and intelligibility of a word sequence.

N-gram models

- an approximation of long sequences using short n-grams
- a straightforward implementation
- an intuitive approach
- good local fluency

Randomly generated text

“Jsi nebylo vidět vteřin přestal po schodech se dal do deníku a položili se táhl ji viděl na konci místnosti 101,” řekl důstojník.

Hungarian

A társaság kötelezettségeiért kapta a középkori temploma az volt, hogy a felhasználók az adottságai, a felhasználó azonosítása az egyesület alapszabályát.

N-gram models, naïve approach

$$W = w_1, w_2, \cdots, w_n$$

$$p(W) = \prod_i p(w_i | w_1 \cdots w_{i-1})$$

Markov's assumption

$$p(W) = \prod_i p(w_i | w_{i-2}, w_{i-1})$$

$$p(\textit{this is a sentence}) = p(\textit{this}) \times p(\textit{is}|\textit{this}) \times p(a|\textit{this}, \textit{is}) \times p(\textit{sentence}|\textit{is}, a)$$

$$p(a|\textit{this}, \textit{is}) = \frac{|\textit{this is a}|}{|\textit{this is}|}$$

Sparse data problem.

Computing, LM probabilities estimation

Trigram model uses 2 preceding words for probability learning. Using **maximum-likelihood estimation**:

$$p(w_3|w_1, w_2) = \frac{count(w_1, w_2, w_3)}{\sum_w count(w_1, w_2, w)}$$

quadrigram: *(lord, of, the, ?)* ()

w	**count**	$p(w)$
rings	30,156	0.425
flies	2,977	0.042
well	1,536	0.021
manor	907	0.012
dance	767	0.010
...		

Large LM – n-gram counts

How many unique n-grams in a corpus?

order	unique	singletons
unigram	86,700	33,447 (38.6%)
bigram	1,948,935	1,132,844 (58.1%)
trigram	8,092,798	6,022,286 (74.4%)
4-gram	15,303,847	13,081,621 (85.5%)
5-gram	19,882,175	18,324,577 (92.2%)

Corpus: Europarl, 30 M tokens.

Language models smoothing

The problem: an n-gram is missing in the data but is in a *sentence* $\rightarrow$ $p(\mathit{sentence}) = 0$.

We need to assign non-zero p for *unseen data*. This must hold:

$$\forall w : p(w) > 0$$

The issue is more pronounced for higher-order models.

Smoothing: an attempt to amend real counts of n-grams to expected counts in any (unseen) data.

Add-one, Add-α, Good–Turing smoothing

Deleted estimation

We can find unseen n-grams in another corpus. N-grams contained in one of them and not in the other help us to estimate general amount of unseen n-grams.

E.g. bigrams not occurring in a training corpus but present in the other corpus million times (given the amount of all possible bigrams equals 7.5 billions) will occur approx.

$$\frac{10^6}{7.5 \times 10^9} = 0.00013\times$$

Interpolation and back-off

Previous methods treated all unseen n-grams the same. Consider trigrams

beautiful young girl
beautiful young granny

Despite we don't have any of these in our training data, the former trigram should be more probable.

We will use probability of lower order models, for which we have necessary data:

young girl
young granny
beautiful young

Interpolation

$$p_I(w_3|w_1w_2) = \lambda_1 p(w_3) + \lambda_2 p(w_3|w_2) + \lambda_3 p(w_3|w_1w_2)$$

If we have enough data we can trust higher order models more and assign a higher significance to corresponding n-grams.

p_I is probability distribution, thus this must hold:

$$\forall \lambda_n : 0 \leq \lambda_n \leq 1$$
$$\sum_n \lambda_n = 1$$

Quality and comparison of LMs

We need to compare quality of various LM (various orders, various data, smoothing techniques etc.)

1) extrinsic (WER, MT, ASR, OCR) and 2) intrinsic (perplexity) evaluation

A good LM should assign a higher probability to a good (looking) text than to an incorrect text. For a fixed test text we can compare various LMs.

Cross-entropy

$$\begin{aligned} H(p_{LM}) &= -\frac{1}{n} \log p_{LM}(w_1, w_2, \ldots w_n) \\ &= -\frac{1}{n} \sum_{i=1}^{n} \log p_{LM}(w_i | w_1, \ldots w_{i-1}) \end{aligned}$$

Cross-entropy is average value of negative logarithms of words probabilities in testing text. It corresponds to a measure of uncertainty of a probability distribution. **The lower the better**.

A good LM should reach entropy close to real entropy of language. That can't be measured directly but quite reliable estimates exist, e.g. Shannon's game. For English, entropy is estimated to approx. 1.3 bit per letter.

Perplexity

$$PP = 2^{H(p_{LM})}$$

Perplexity is a simple transformation of cross-entropy.

A good LM should not waste p for improbable phenomena.

The lower entropy, the better $\rightarrow$ the lower perplexity, the better.

Comparing smoothing methods (Europarl)

method	**perplexity**
add-one	382.2
add-α	113.2
deleted est.	113.4
Good–Turing	112.9