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

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

NRU ITMO, St Petersburg, 2019
anton.m.alexeyev+itmo@gmail.com

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

Markov property

N-gram models we discussed earlier actually are **Markov models**

Markov property: conditional distribution of the next state of a stochastic process depends only on current state

$$\mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = \mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n)$$

The process with discrete time (or a sequence of random events), that has this property is called a **Markov chain**

A simple and a well-studied probabilistic model suitable for many tasks

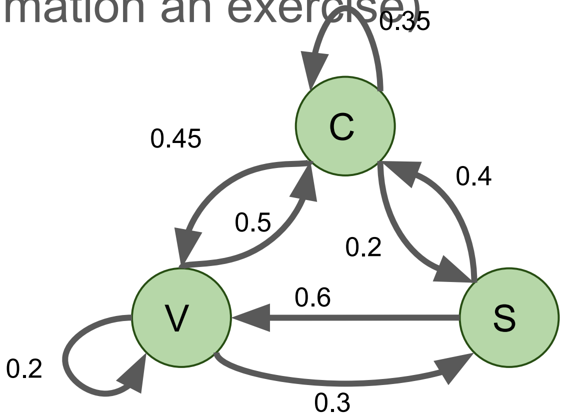
Markov chain

The model is entirely set by the stochastic matrix = transitions probabilities matrix

Example. Events: vowel (v), consonant (c), white**s**pace/punctuation (s) (probabilities are set at random, consider the estimation an exercise)

P_{trans} =

	v	c	s
v	0.2	0.5	0.3
c	0.45	0.35	0.2
s	0.6	0.4	0.0



DEMO: ugly self-promotion: http://antonalexeev.hop.ru/markov/index.html

Markov chains

- So — Markov chain as a process is set by the matrix of transitions probabilities and probabilities of initial states

$$\pi = (p_1^{(0)}, ..., p_n^{(0)})^T$$

$$P_{trans} = \{p_{i\to j},\ i, j \in 1:n, \sum_{j=1}^{n} p_{i\to j} = 1 \forall i\}$$

- Probability of a trajectory
  of length one x_i

$$p = p_i$$

  of length two $x_i \to x_j$

$$p = p(x_i)p(x_j|x_i) = \pi_i P_{i,j}$$

  of length three $x_i \to x_j \to x_k$

$$p = p(x_i)p(x_j|x_i)p(x_k|x_i, x_j) = p(x_i)p(x_j|x_i)p(x_k|x_j) = \pi_i P_{i,j} P_{j,k}$$

Markov chains

- 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

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

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

and

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

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

Stationary distribution: interpretation

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 №1 (previous lecture)

N-gram model

- Model:

$$P(x_1, \dots x_n) = \prod_{i=1}^{n} P(x_i | x_{i-N+1} \dots x_{i-1})$$

one has to add $N-1$ terms «begin» ^and «end» $ from both sides (padding)

- We can estimate the probability like that

$$P(x_i | x_{i-N+1} \dots x_{i-1}) = \frac{Count(x_{i-N+1} \dots x_{i-1} x_i)}{Count(x_{i-N+1} \dots x_{i-1})}$$

- $$P(x_i | x_{i-1}) = Count(x_i, x_{i-1}) Count(x_{i-1})$$

- E.g. for bigrams:

$$P(hello, i, love, you) =$$

$$= P(hello|\hat{}\,)P(i|hello)P(love|i)P(you|love)P(\$|you)$$

Application example №2: PageRank

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

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

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

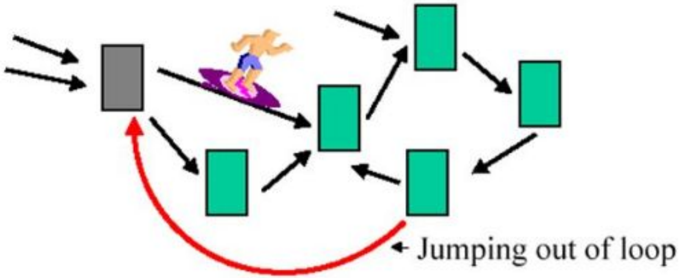
PageRank is given by a small modification of BrokenRank:

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

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

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

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

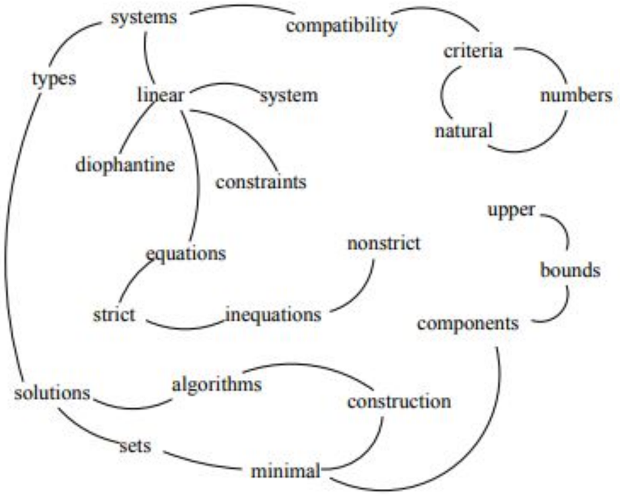
Application example №3: PageRank (TextRank)

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

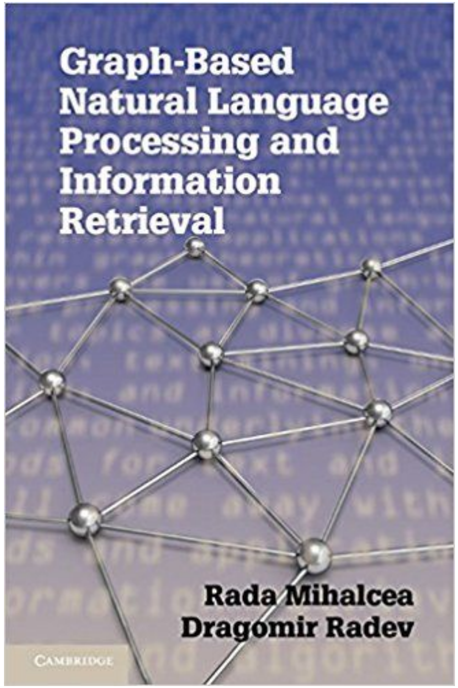
Keywords assigned by human annotators:
linear constraints; linear diophantine equations; minimal generating sets; non-strict inequations; set of natural numbers; strict inequations; upper bounds

Mihalcea, R., Tarau, P. TextRank: Bringing Order into Texts. // Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing. — 2004. — Vol. 4. — № 4. — P. 404–411

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



Other applications

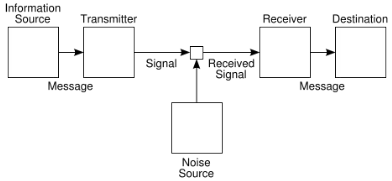
- Language detection
- Named-entity recognition
- POS-tagging
- Speech recognition
- …useful almost every time we deal with sequences

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°

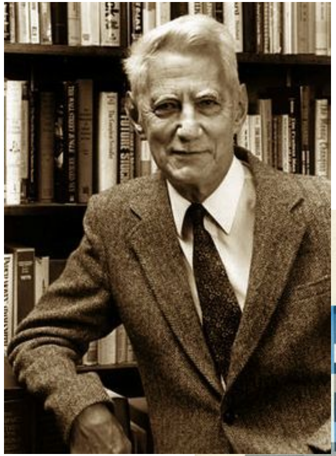


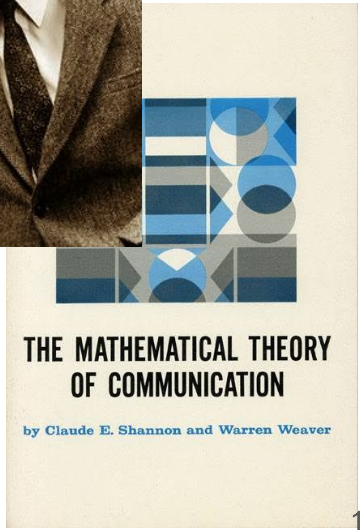
1948 - A Mathematical Theory of Communication,
Claude Shannon; information theory foundations are introduced

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.





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: $p(x_i) = \frac{1}{N} \;\; \forall x \in 1 : N$

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

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

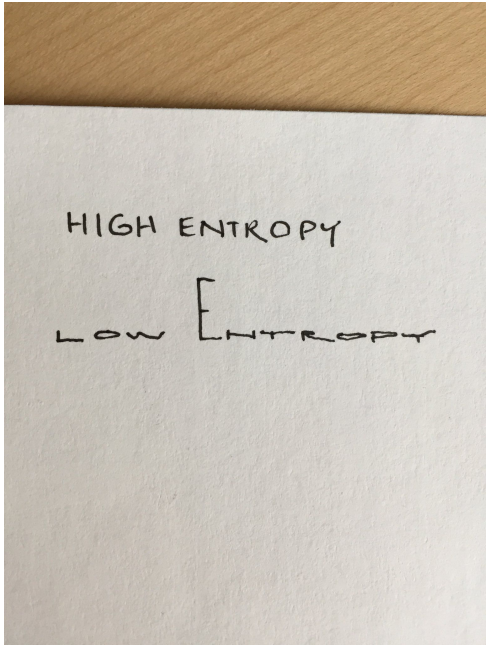
1. ~~Markov chains~~
    a. ~~Language models~~
    b. ~~Keywords extraction and other applications~~

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

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

$$\Updownarrow$$

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

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

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

NRU ITMO, St Petersburg, 2019
anton.m.alexeyev+itmo@gmail.com