processamento de língua natural

modelos de linguagens

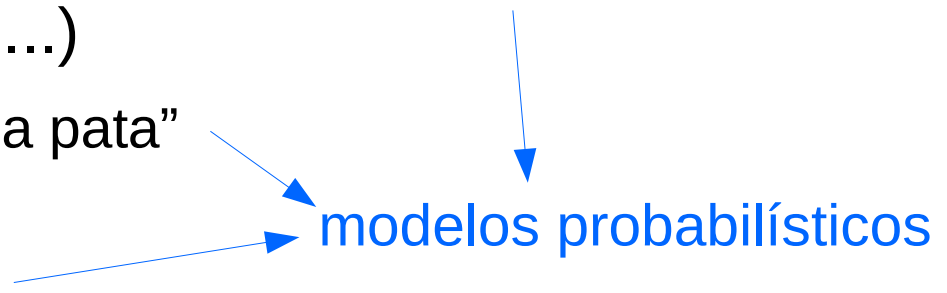
todas as linguagens têm modelos

em linguagens formais – Java, python – os modelos são precisos → semântica clara

em língua natural há regras gramaticais, mas a formalidade não é tão forte (poesia, prosa, ...)

e há **ambiguidade**, ex. “ele viu a pata”

língua natural é muito variada



modelos *n*-grama de letras

n-grama (em geral): sequências de *n* símbolos

com letras:

- unigrama → probabilidades de letras (*n*=1)
- bigrama → probabilidades de sequências de 2 letras (*n*=2)
- trigrama → probabilidades de sequências de 3 letras (*n*=3)

…

- *n*-grama → probabilidades de sequências de *n* letras: $P(c_{1:N})$

modelo *n*-grama

é uma cadeia de Markov de ordem *n*-1

cadeia de Markov: a probabilidade de um estado só depende do anterior

(de ordem *n*: só depende dos *n* estados anteriores)

bigrama $P(c_{1:N}) = \prod_{i=1}^{N} P(c_i|c_{1:i-1}) = \prod_{i=1}^{N} P(c_i|c_{i-1})$

cadeia de Markov de ordem 1

trigrama $P(c_{1:N}) = \prod_{i=1}^{N} P(c_i|c_{1:i-1}) = \prod_{i=1}^{N} P(c_i|c_{i-2:i-1})$

cadeia de Markov de ordem 2

n-gramas de letras em números

para uma língua com 26 carateres são teoricamente possíveis

676 bigramas

15.576 trigramas

456.976 tetragramas

mas numa língua real nem todos existem

- na realidade existem muito poucos – **conjunto esparso**

n-gramas em palavras

em inglês

```
File sizes: approx. 24 GB compressed (gzip'ed) text files

Number of tokens:     1,024,908,267,229
Number of sentences:     95,119,665,584
Number of unigrams:          13,588,391
Number of bigrams:          314,843,401
Number of trigrams:         977,069,902
Number of fourgrams:      1,313,818,354
Number of fivegrams:      1,176,470,663
```

corpus

estimativa a partir de

conjunto muito esparso!

http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

identificação de línguas com modelos *n*-grama (de letras)

ex. construir um modelo trigrama de cada língua candidata

$$P(c_i | c_{i-2:i-1}, l)$$

estimativas obtidas a partir de *corpora* de cada língua

~100.000 letras necessárias

obtém-se $\mathbf{P}(Texto | Língua)$

aplicando a regra de Bayes →

$$\begin{aligned} l^* &= \underset{l}{\operatorname{argmax}}\, P(l | c_{1:N}) \\ &= \underset{l}{\operatorname{argmax}}\, P(l) P(c_{1:N} | l) \\ &= \underset{l}{\operatorname{argmax}}\, P(l) \prod_{i=1}^{N} P(c_i | c_{i-2:i-1}, l) \end{aligned}$$

n-gramas de letras

úteis também para

- correção de erros
- classificação de tipo de doc. (legal, científico, notícia,…)
- reconhecimento de entidades (classes a que pertencem)

Sr. Elidérico comprou cipralex
pessoa medicamento

modelos de letras são preferíveis nestes casos

- permitem associar “ex ” com um medicamento
  e “Sr. *” com uma pessoa

n-gramas de palavras

ex. a partir de modelos criados com o corpus AIMA, fazendo depois, em cada caso, uma série de amostragens aleatórias

unigram: logical are as are confusion a may right tries agent goal the was . . .

bigram: systems are very similar computational approach would be represented . . .

trigram: planning and scheduling are integrated the success of naive bayes model is . . .

recuperação de informação (*information retrieval* - IR)

o cerne dos motores de pesquisa!

componentes

1. corpus de documentos
2. interrogações numa linguagem definida
3. conjunto de resultados – relevância
4. apresentação dos resultados

IR estatística

fatores de avaliação da relevância, num corpus com *N* docs

1. *term frequency* (*TF*)

   frequência com que um termo da pergunta, q_i, aparece num doc.

2. *inverse document frequency* (*IDF*)

   inverso do nº de docs. em que um termo da pergunta aparece

3. *comprimento do documento*

   no caso do doc. *i* o comprimento é $|d_i|$

   e o comprimento médio dos documentos no corpus é $L = \sum_i |d_i| / N$

princípios norteadores

um documento que mencione várias vezes um termo procurado tem uma relevância elevada

um termo da procura que aparece em muitos documentos tem uma relevância reduzida

entre dois documentos com todos os termos procurados, o menor deles tem mais relevância

função de avaliação para IR

ex. a *BM25* é uma forma da *TF-IDF*

$$BM25(d_j, q_{1:N}) = \sum_{i=1}^{N} IDF(q_i) \frac{TF(q_i, d_j)(k+1)}{TF(q_i, d_j) + k\left(1 - b + b\frac{|d_j|}{L}\right)}$$

em que *b* e *k* são parâmetros que podem ser ajustados com validação cruzada

valores típicos: $b = 0{,}75$ e $k = 2{,}0$

sendo a *IDF* dada por

$$IDF(q_i) = \log \frac{N - DF(q_i) + 0{,}5}{DF(q_i) + 0{,}5}$$

*BM*25 implementação

$DF(q_i)$ é o nº de documentos no corpus que têm o termo q_i

o sistema de IR cria um índice que lista, para cada palavra do vocabulário, os documentos que contêm essa palavra, ***hit list***

a resposta a uma pergunta calcula a interseção das *hit lists* de todos os termos da pergunta

só se avaliam os documentos que resultam da interseção

avaliação dos resultados de IR

suponhamos um conjunto de peritos que faz uma pergunta sobre um corpus de 100 documentos e sabe quais os documentos relevantes

o resultado do IR:

classificação do sistema IR

documentos	no resultado	não no resultado
relevantes	30	20
não relevantes	10	40

classificação dos peritos

falsos negativos

falsos positivos

define-se

precisão=(relevantes retornados)/(total retornados), i.e. 30/(30+10)=0,75

cobertura (*recall*)=(relevantes retornados)/(total relevantes), 30/(30+20)=0,6

os limites de precisão e cobertura

pode aumentar-se artificialmente a precisão à custa da cobertura e vice-versa!

- no limite, retornando todos os documentos ⇒ cobertura = 100%
  mas a precisão será baixa
- ou, retornando um único documento relevante ⇒ precisão = 100%
  mas a cobertura será baixa

na pesquisa na *web*

- estima-se a cobertura (não é viável avaliar todos os documentos)
- ou usa-se só precisão, e apenas para os *N* mais relevantes,
  ex. P@10 ou P@50

medida combinada

uma maneira habitual de combinar a relevância de

precisão

e

cobertura

é a medida F_1 definida por

$$F_1 = \frac{2 \times precisão \times cobertura}{precisão + cobertura}$$

média harmónica entre precisão e cobertura

IR com mais qualidade

usar as raízes das palavras (perguntas podem ser feitas de forma genérica)

- **lematização** (*lemmatisation*)
  - ex: lema de “gostei” é “gostar”
- **redução ao radical** (*stemming*)
  - ex: radical de “gostei” é “gost”

usar sinónimos

- através de dicionários, ou de relações entre perguntas

usar metadados

- ex. hiperligações

o *PageRank*

uma das ideias originais da Google

dá mais relevância a páginas que são muito referidas por várias outras (*in-links*)

evitando manipulações: dá mais peso às ligações de páginas mais relevantes (que têm mais *in-links*)

definição recursiva

$$PR(p) = \frac{1-d}{N} + d \sum_i \frac{PR(\text{in}_i)}{C(\text{in}_i)}$$

N: total de páginas no corpus
in_i: páginas que ligam à pág. *p*
$C(\text{in}_i)$: # links na pág. in_i
d: fator de amortecimento

começa com todas as páginas com $PR(p)=1$

e itera atualizando os *rankings* até convergirem

outros aspetos (ILN - MEI)

análise gramatical

- sintática

  identificação dos componentes de uma frase

  sujeito, predicado, complemento direto, …

- léxica

  análise sintática necessita da definição de categorias léxicas

  substantivos, verbos, pronomes, advérbios, …

  e dicionários com as palavras respetivas

interpretação semântica

resultados interessantes com aprendizagem em programação lógica indutiva

outras vertentes de PLN

- extração de informação
- tradução automática
- resumos
- diálogo

  conhecimento contextual

  a aprendizagem profunda tem dado muitos resultados recentemente

  analisando palavras / frases no seu contexto – *embeddings*