MCZA017-13
Processamento de Linguagem Natural

Semântica e similaridade de palavras: Parte III: Matriz termo-contexto

Prof. Jesús P. Mena-Chalco
jesus.mena@ufabc.edu.br

2Q-2019

Da aula anterior...

Matriz de co-ocorrência: termo-documento

Dois **documentos** são similares (semantica) se os vetores são similares

V

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

A dimensão do vetor é o tamanho do vocabulário: $N^{|V|}$

Matriz de co-ocorrência: termo-documento

Duas **palavras** são similares (semantica) se os vetores são similares

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

A dimensão do vetor é o número de documentos: $N^{|D|}$

Abordagem simples mas quais seriam os problemas?

Matriz: termo-documento

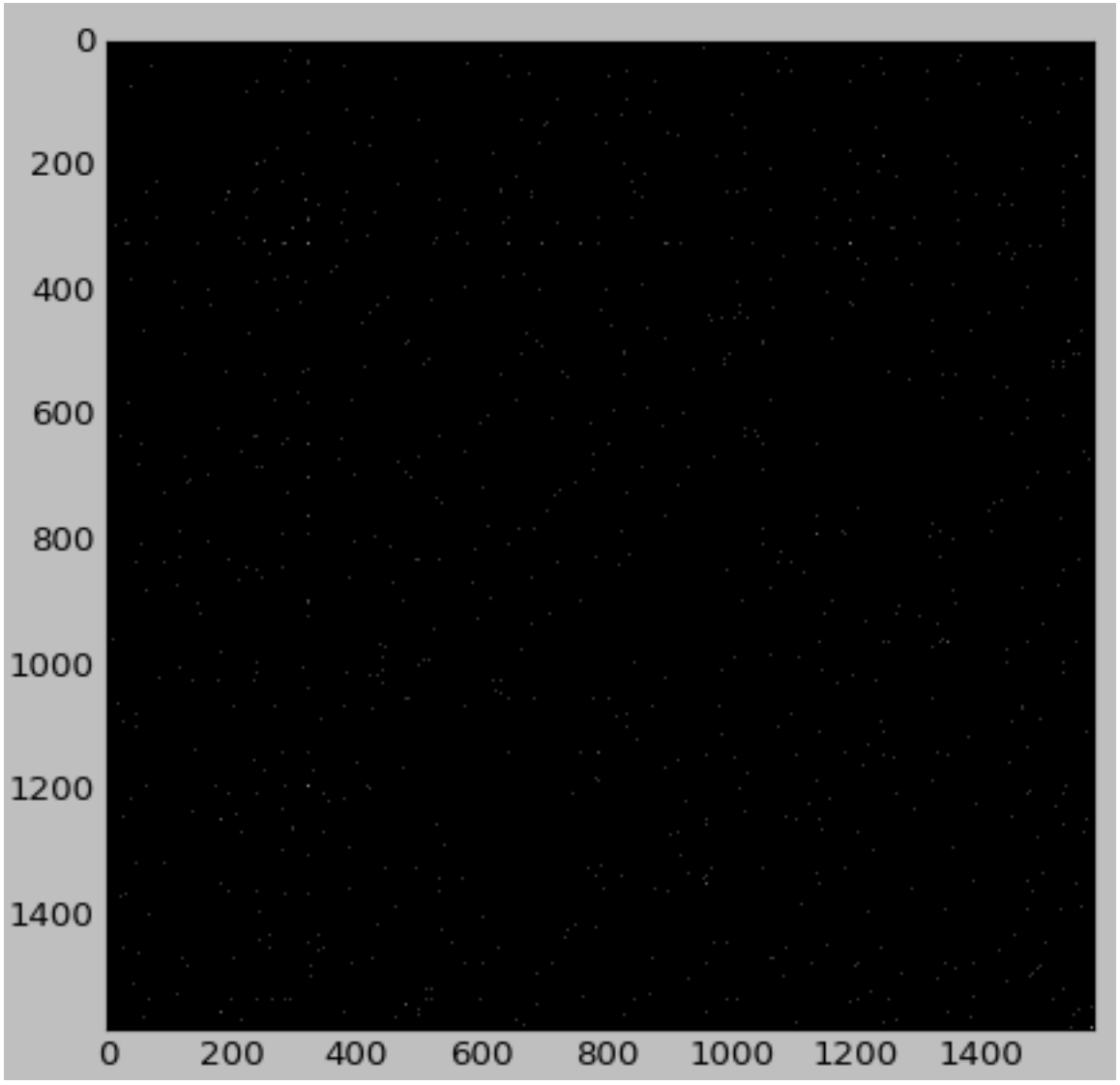
- A similaridade entre palavras considera **todas as palavras** presentes em todos os documentos.
- Os vetores tem muitos elementos nulos (**vetores esparsos**)
- O tamanho do vetor **depende** do número de documentos.

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

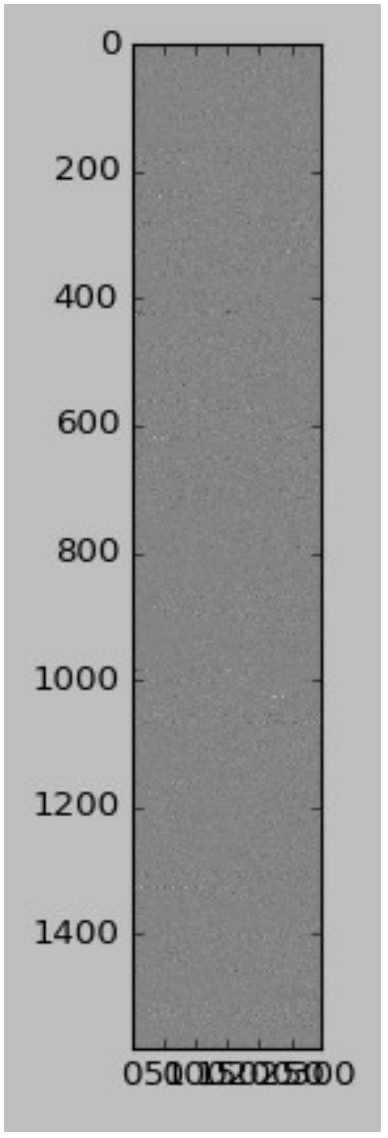
Matriz esparsa

$$\begin{bmatrix} 0 & 0 & 3 & 0 & 4 \\ 0 & 0 & 5 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 6 & 0 & 0 \end{bmatrix}$$

Matriz esparsa

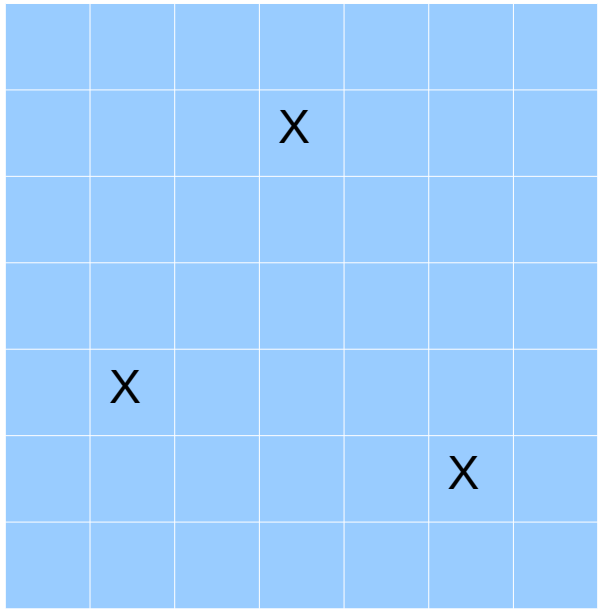


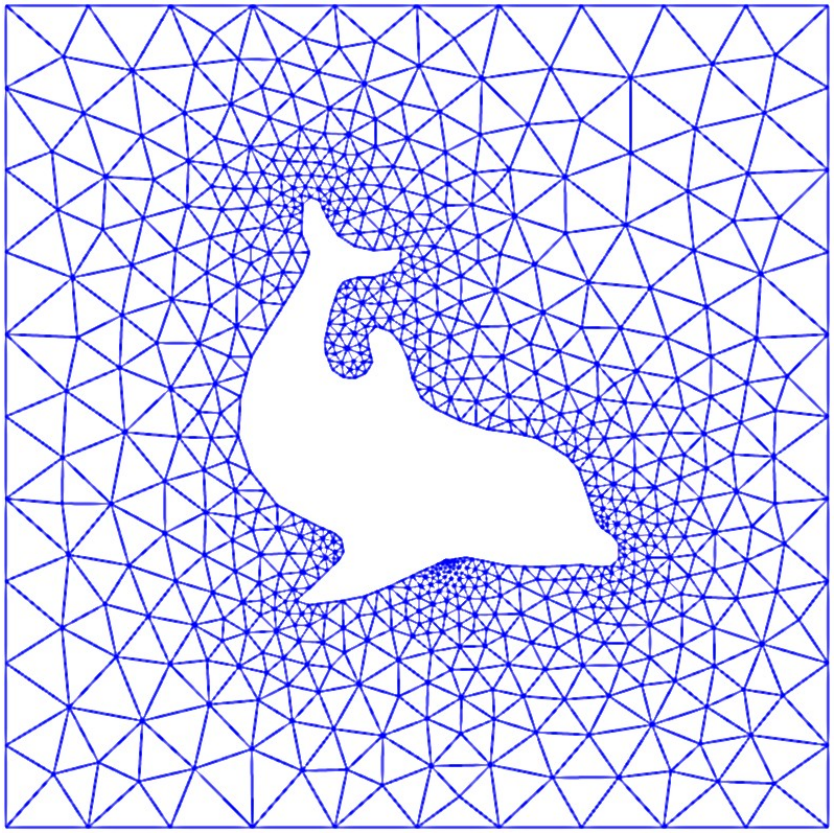
Dimensão |V| x |D|



Dimensão |V| x |D|

Uso de estruturas de dados mais sofisticadas ...





Matriz termo-documento

- O mais natural seria considerar:
  - Matriz termo-termo.
  - Matriz palavra-palavra.
  - Matriz **termo-contexto**.

Sinônimos

- Dimensão |V| x |V|

- Cada celula da matriz registra o número de vezes que a palavra (da linha) co-ocorre com outra palavra (da coluna) **em um contexto**, dado em um corpus de treinamento.

O contexto poderia ser o documento:
Cada celula representa o número de vezes que duas palavras estão presentes no documento

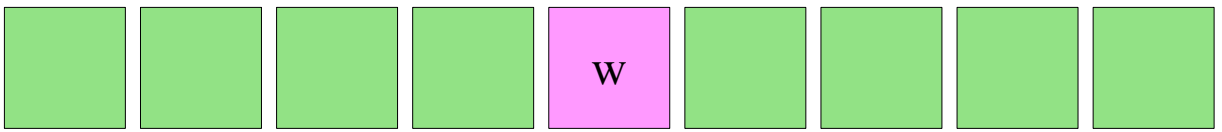
Matriz termo-contexto

- Por que não usar um **contexto** menor?

Obras similares estão "geralmente" próximas.

Palavras que estão em contextos similares, tendem a ser semanticamente similares

- Por exemplo: parâgrafos.
- Por exemplo: usar 4 palavras antes e depois de uma determinada palavra.



Matriz termo-contexto

Exemplo de contexto local de palavras:

- O carro é rápido?
  Sim, ele é muito veloz!

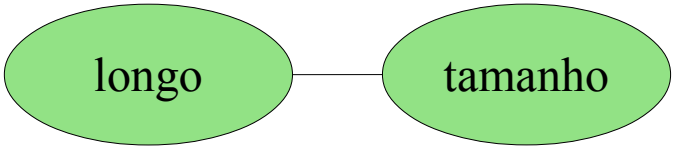
- O manuscrito é longo.
  Geralmente o tamanho é menor.

Matriz termo-contexto

Exemplo de contexto local de palavras:

- O carro é rápido?
  Sim, ele é muito veloz!

- O manuscrito é longo.
  Geralmente o tamanho é menor.



Quatro exemplos: Corpus Brown

- **sugar**, a sliced lemon, a tablespoonful of **apricot** preserve or jam, a **pinch** each of,
- their enjoyment. Cautiously she sampled her first **pineapple** and another **fruit** whose taste she likened
- well suited to **programming** on the **digital computer**. In finding the optimal R-stage policy from
- for the purpose of gathering **data** and **information** necessary for the study authorized in the

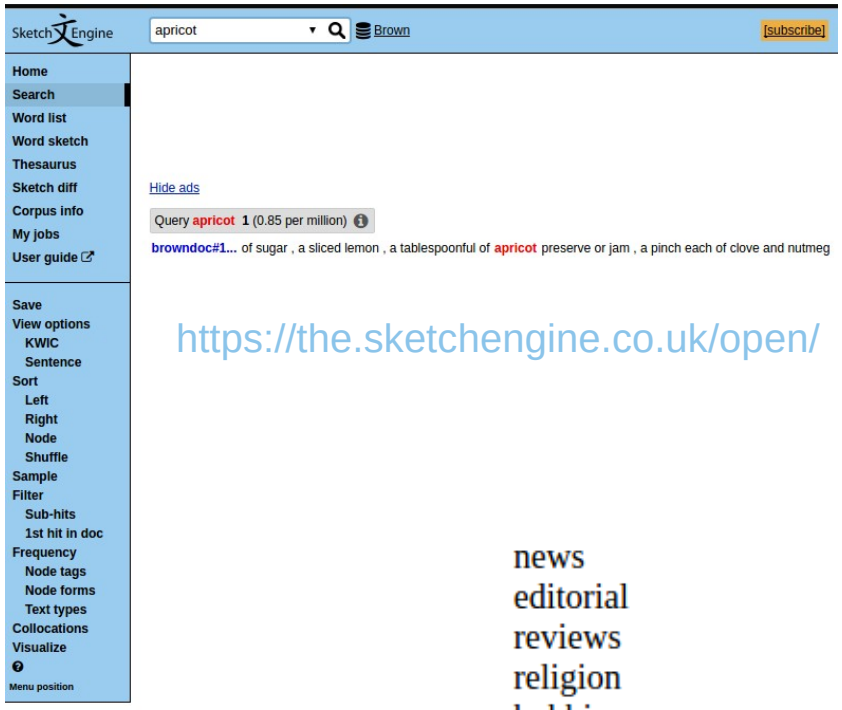
Corpus Brown: 1960

The Brown University Standard Corpus of Present-Day American English

Criado na decada dos **1960** é um corpus geral que contem 500 amostras em inglês.

Ao todo 1,014,312 palavras de trabalhos publicacos nos Estados Unidos em 1961 (500 fontes), organizado em **15** categorias.

É um corpus **pioneiro** na área de linguística computacional.



https://the.sketchengine.co.uk/open/

news
editorial
reviews
religion
hobbies
lore
belles_lettres
government
learned
fiction
mystery
science_fiction
adventure
romance
humor

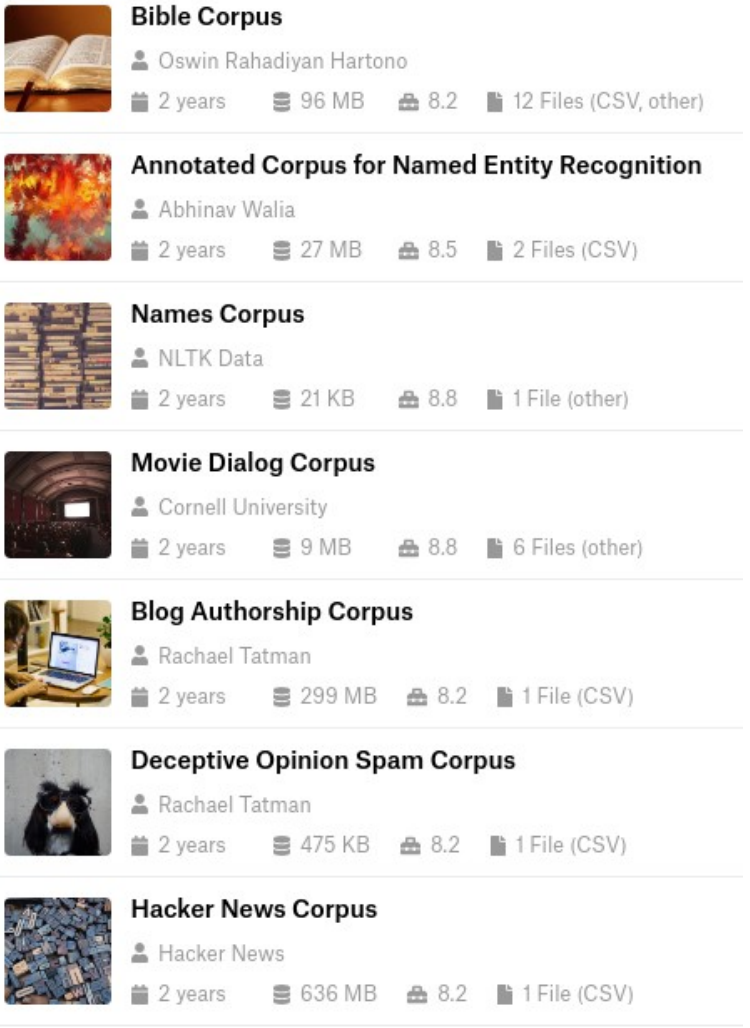
Corpus Brown: 1960

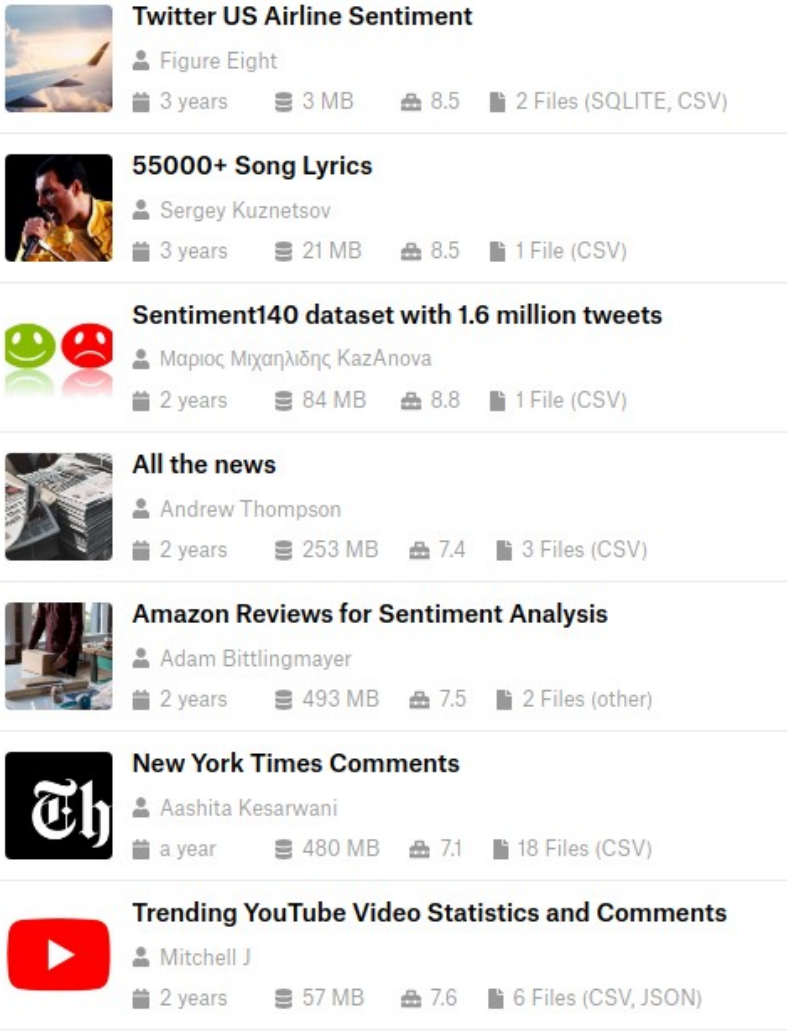
O corpus está composto de 500 textos, cada um contendo mais de 2000 palavras:

1) PRESS: REPORTAGE (44 texts)
2) PRESS: EDITORIAL (27 texts)
3) PRESS: REVIEWS (17 texts)
4) RELIGION (17 texts)
5) SKILL AND HOBBIES (36 texts)
6) POPULAR LORE (48 texts)
7) BELLES-LETTRES (75 texts)
8) MISCELLANEOUS: GOVERNMENT & HOUSE ORGANS (30 texts)
9) LEARNED (80 texts)
10) FICTION: GENERAL (29 texts)
11) FICTION: MYSTERY (24 texts)
12) FICTION: SCIENCE (6 texts)
13) FICTION: ADVENTURE (29 texts)
14) FICTION: ROMANCE (29 texts)
15) HUMOR (9 texts)

Outros corpora

https://www.kaggle.com/datasets?search=corpus





Quatro exemplos: Corpus Brown

+- 7 palavras

- sugar, a sliced lemon, a tablespoonful of **apricot** preserve or jam, a pinch each of,
- their enjoyment. Cautiously she sampled her first **pineapple** and another fruit whose taste she likened
- well suited to programming on the digital **computer**. In finding the optimal R-stage policy from
- for the purpose of gathering data and **information** necessary for the study authorized in the

Matriz termo-contexto (+-7 palavras)

Uma palavra é representada por um vetor de números que **consideram um contexto**

	aardvark	...	computer	data	pinch	result	sugar	...
apricot	0	...	0	0	1	0	1	
pineapple	0	...	0	0	1	0	1	
digital	0	...	2	1	0	1	0	
information	0	...	1	6	0	4	0	

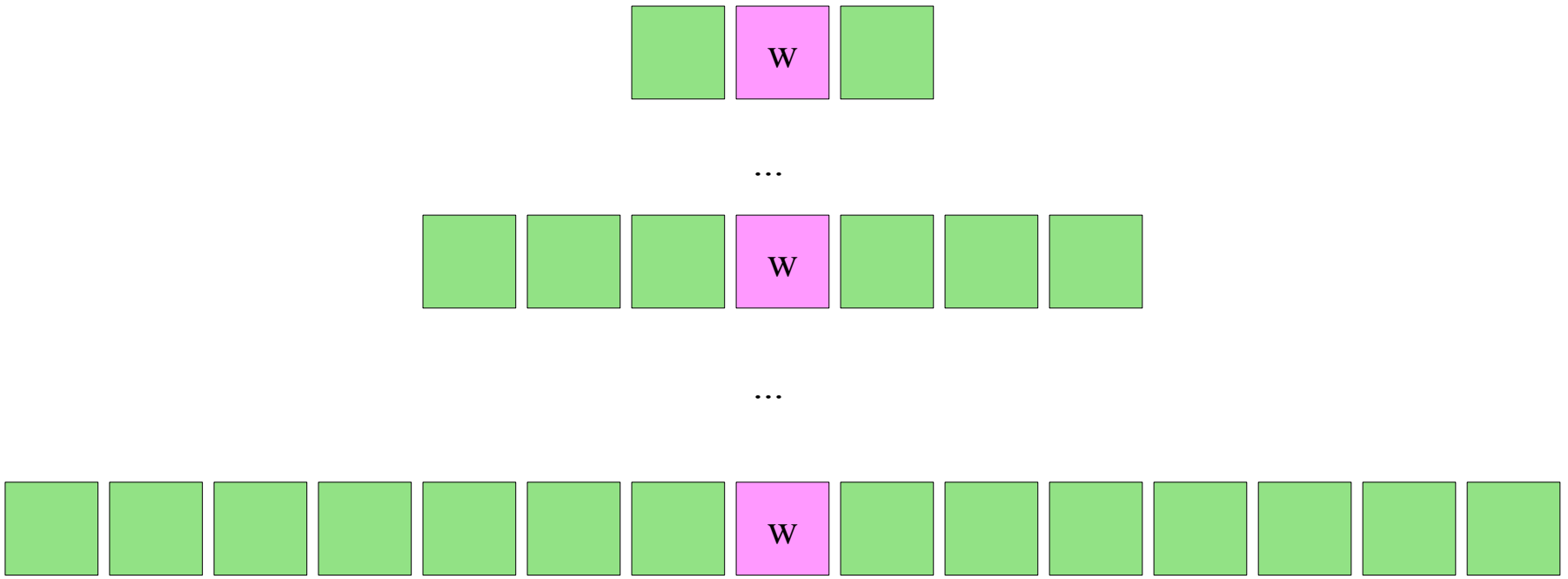
No exemplo temos um vetor de tamanho 6,
mas a dimensão é proporcional ao tamanho do vocabulário considerado

Apricot = [0, 0, 0, 1, 0, …, 3, 0, 0, 0, 0, 0, 1, 1, …, 0, 0, 0, …]

A estrutura de dados deve ser a mais eficiente possível para armazenar um vetor esparso de tamanho ~**20 ou 50 mil** elementos.

Matriz termo-contexto: Janelas de tamanho 1, 3, ..., 7

Similaridade **mais sintática**



Similaridade **mais semântica**

Contexto total de 15 palavras

Teste1.py

```
Document   = dict([])
Vocabulary = set([])

# leitura das stopwords
Stopwords        = set([])
for s in open("stopwords-pt.txt",'r').readlines()
    Stopwords.add(s.strip().lower())

# leitura dos documentos
for fileName in os.listdir(dirDB):
    Document[fileName] = []
    document = open(dirDB+"/"+fileName,'r')
    content  = document.read().lower()

    for w in re.findall(regex, content):
        if w not in Stopwords and len(w)>=3:
            Document[fileName].append(w)
    Vocabulary.update( Document[fileName] )


D = len(Document)
V = len(Vocabulary)
S = len(Stopwords)
```

Teste1.py

```
# contabilizando os pares de palavras
k           = 3
Mcontext    = numpy.zeros((V, V))
iVocabulary = dict([])

for (i,w) in enumerate(Vocabulary):
    iVocabulary[w] = i

for d in Document.keys():
    print (d)
    for (i,w) in enumerate(Document[d]):
        context = []
        if i>k:
            context += Document[d][i-k:i]
        if i<len(Document[d])-k:
            context += Document[d][i+1:i+k+1]

        print (i, w, context)

        iw = iVocabulary[w]
        for wc in context:
            Mcontext[iw, iVocabulary[wc]] += 1
```

notícias

Noticia-Fapesp	atividades humanas já danificaram 75% d
Noticia-Fapesp	fapesp e finep apoiarão pesquisas em qu
Noticia-Fapesp	instituto oceanográfico da usp tem duas o
Noticia-Folha	esquerda critica netflix por causa de série
Noticia-Folha	uma defesa do facebook empresa fracass
Noticia-Sensacionalista	fifa pode suspender jogador que atribuir g
Noticia-Sensacionalista	brasil enfrenta epidemia de arrepios na es

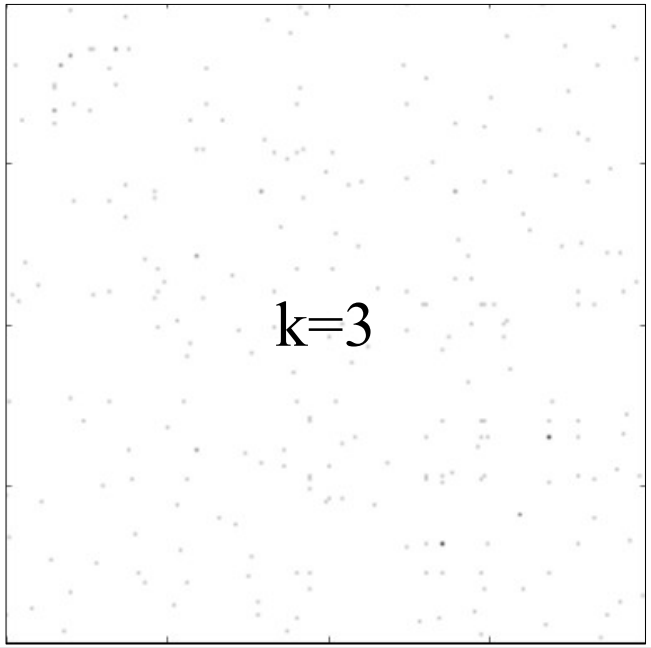
Teste1.py - sete notícias

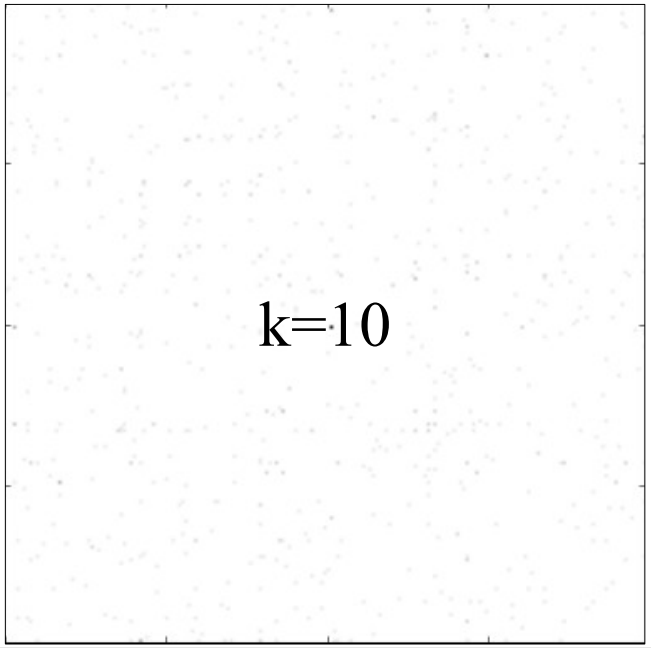
```
118 ser ['trapaça', 'esportiva', 'pode', 'engano', 'jogador', 'alguns']
119 engano ['esportiva', 'pode', 'ser', 'jogador', 'alguns', 'simplesmente']
120 jogador ['pode', 'ser', 'engano', 'alguns', 'simplesmente', 'talentosos']
121 alguns ['ser', 'engano', 'jogador', 'simplesmente', 'talentosos', 'bons']
122 simplesmente ['engano', 'jogador', 'alguns', 'talentosos', 'bons', 'pontaria']
123 talentosos ['jogador', 'alguns', 'simplesmente', 'bons', 'pontaria', 'têm']
124 bons ['alguns', 'simplesmente', 'talentosos', 'pontaria', 'têm', 'muita']
125 pontaria ['simplesmente', 'talentosos', 'bons', 'têm', 'muita', 'tempo']
126 têm ['talentosos', 'bons', 'pontaria', 'muita', 'tempo', 'disse']
127 muita ['bons', 'pontaria', 'têm', 'tempo', 'disse', 'miller']
128 tempo ['pontaria', 'têm', 'muita']
129 disse ['têm', 'muita', 'tempo']
130 miller ['muita', 'tempo', 'disse']


Numero de documentos  : 7
Tamanho do vocabulario: 1580
Numero de stopwords:    212
```

Teste1.py

```
# Informacoes basicas
print("Numero de documentos   : {}".format( D ))
print("Tamanho do vocabulario: {}".format( V ))
print("Numero de stopwords:    {}".format( S ))
plt.imshow(Mcontext[1:200,1:200], cmap='binary')
plt.show()
```

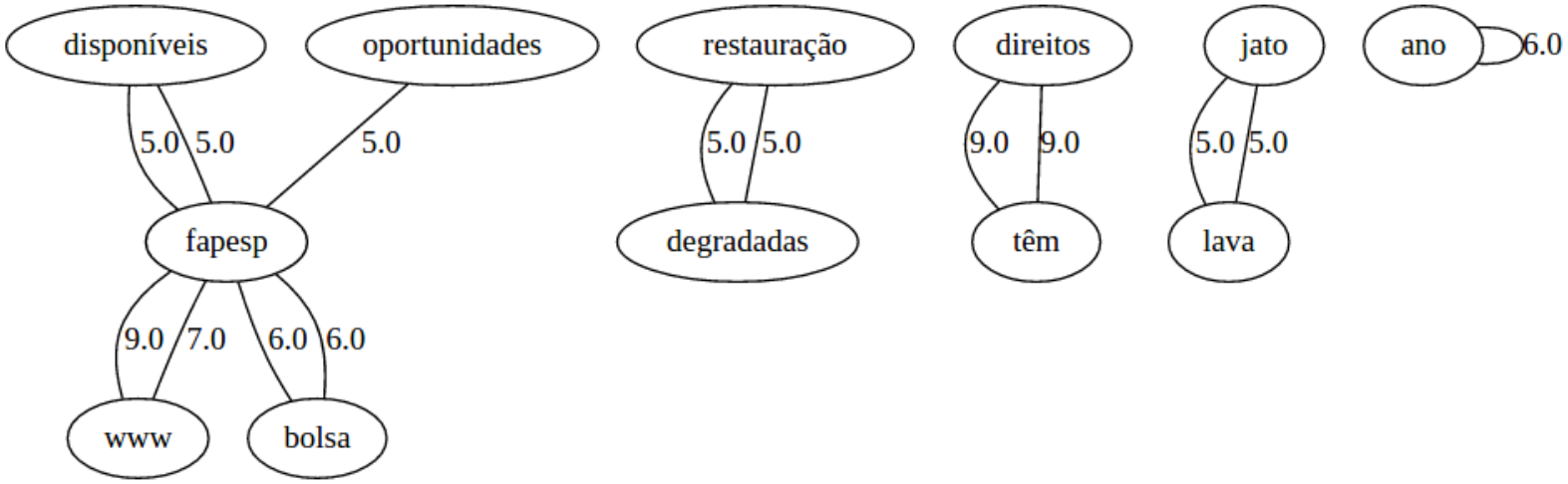
```
python3 teste1.py noticias/
```





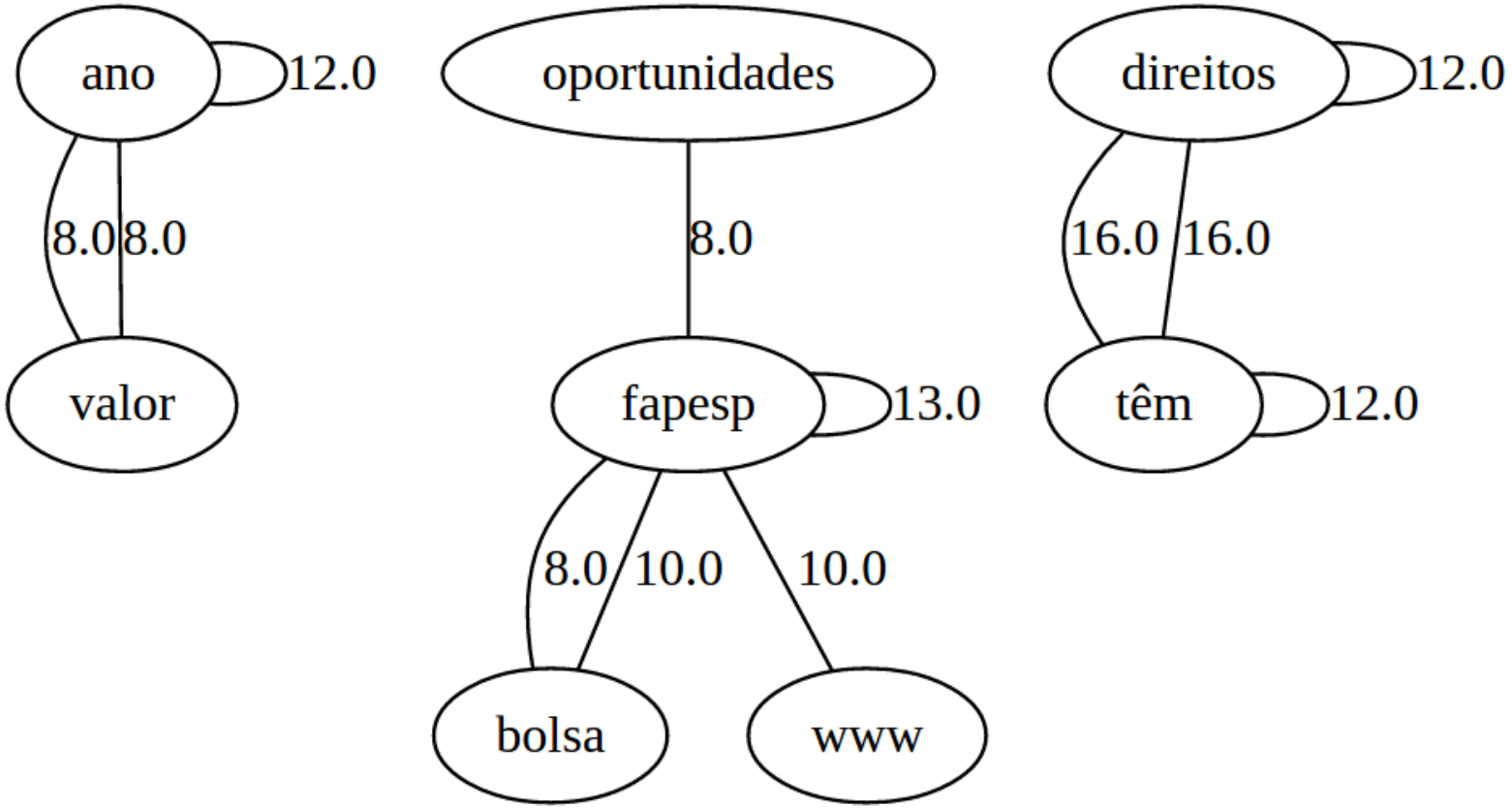
Teste2.py (k=3)

```
python3 teste2.py noticias/
```



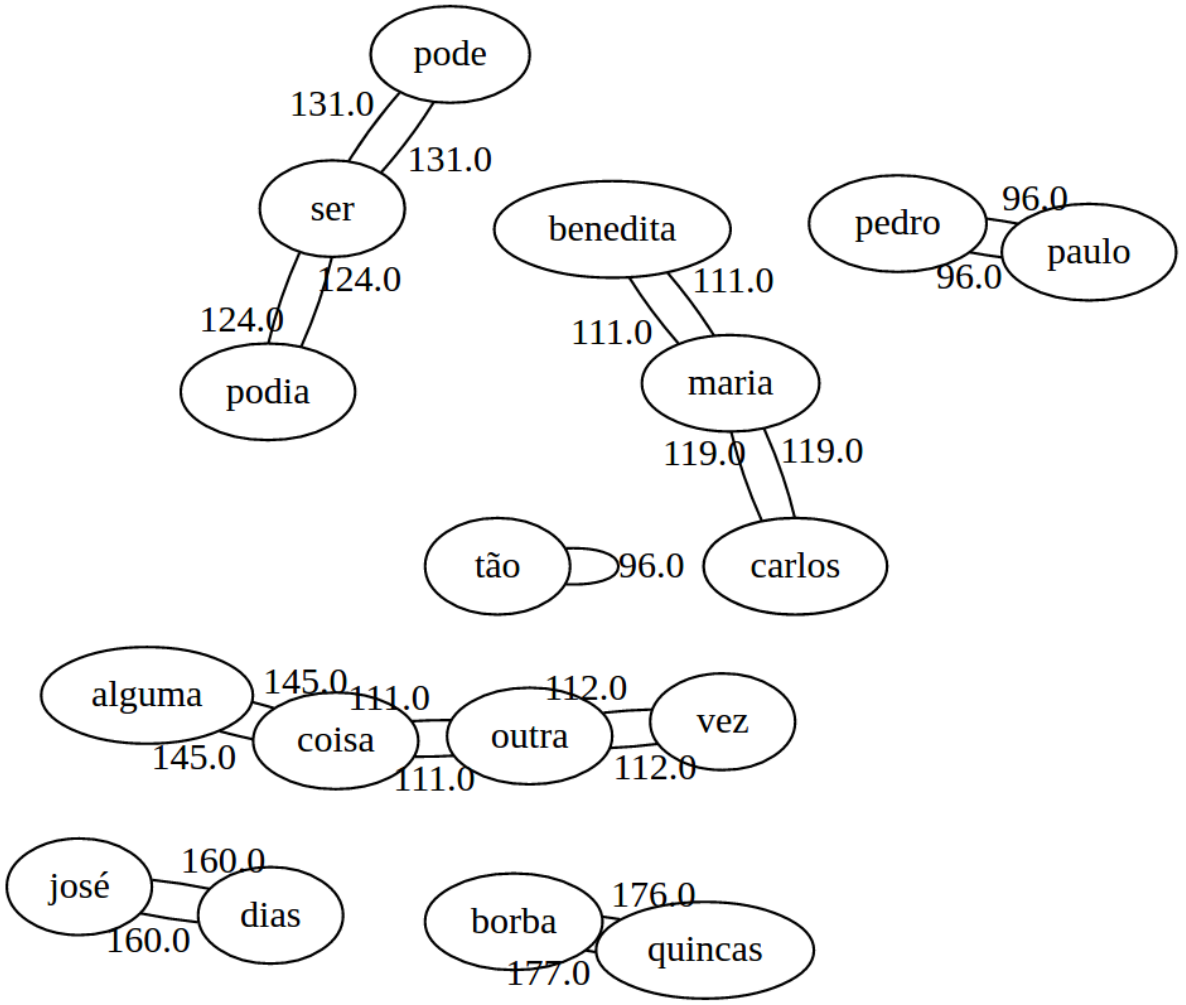
Teste2.py (k=10)

```
python3 teste2.py noticias/
```



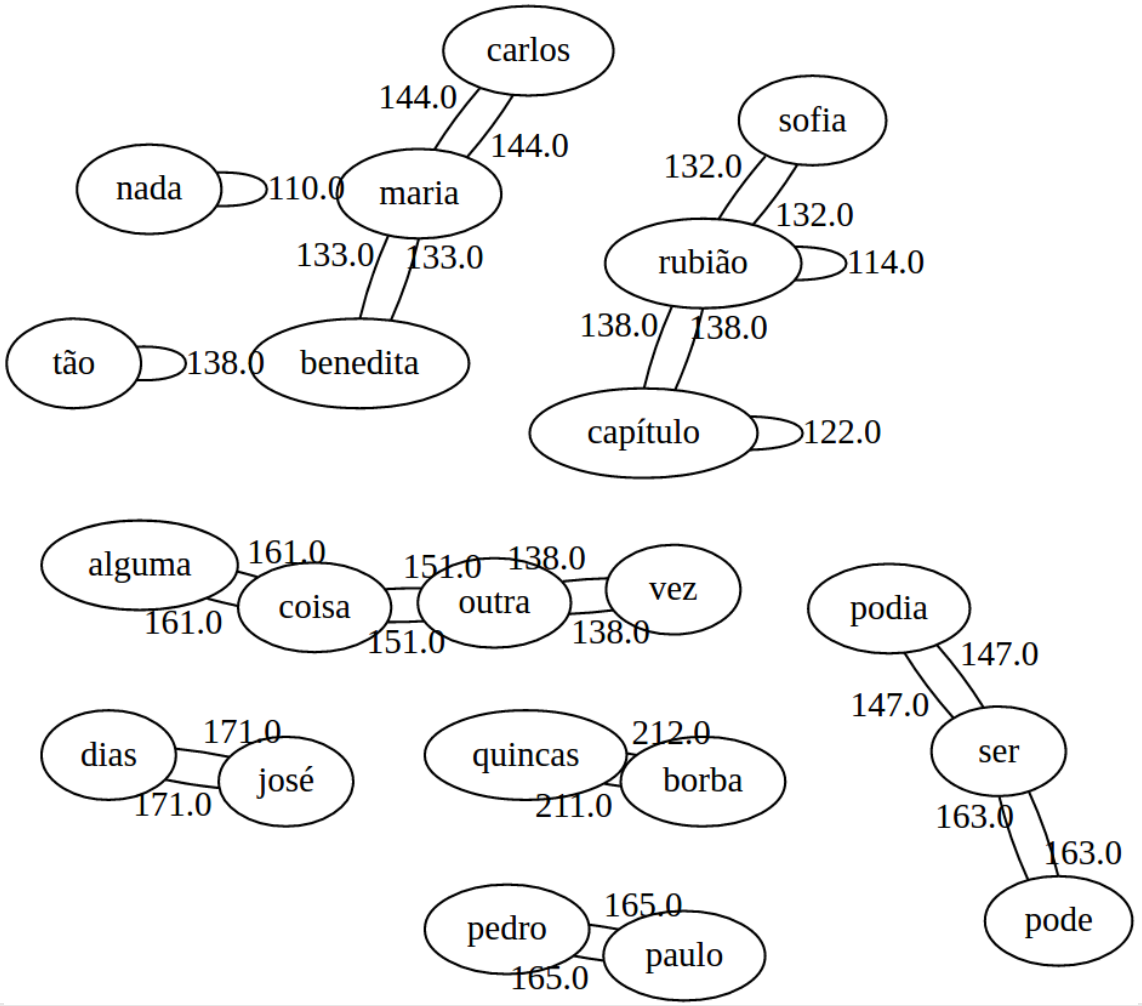
Teste2.py (k=3)

```
python3 teste2.py machado-db/
```



Teste2.py (k=10)

```
python3 teste2.py machado-db/
```



Similaridade baseada em distribuição de palavras

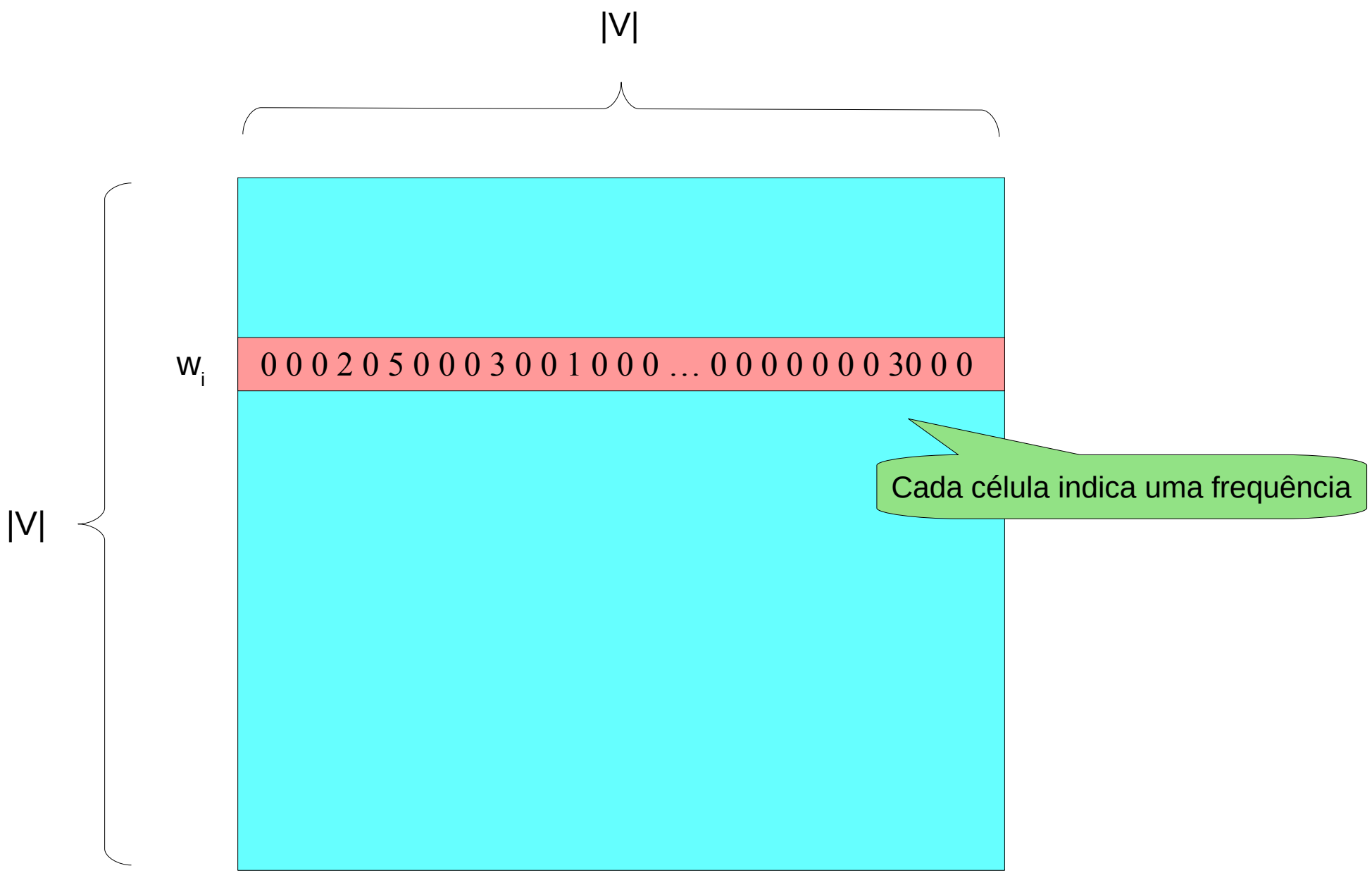
Na literatura isso é conhecido como:

- *Distributional semantics.*
- *Vector semantics.*
- *Vector-space semantics.*

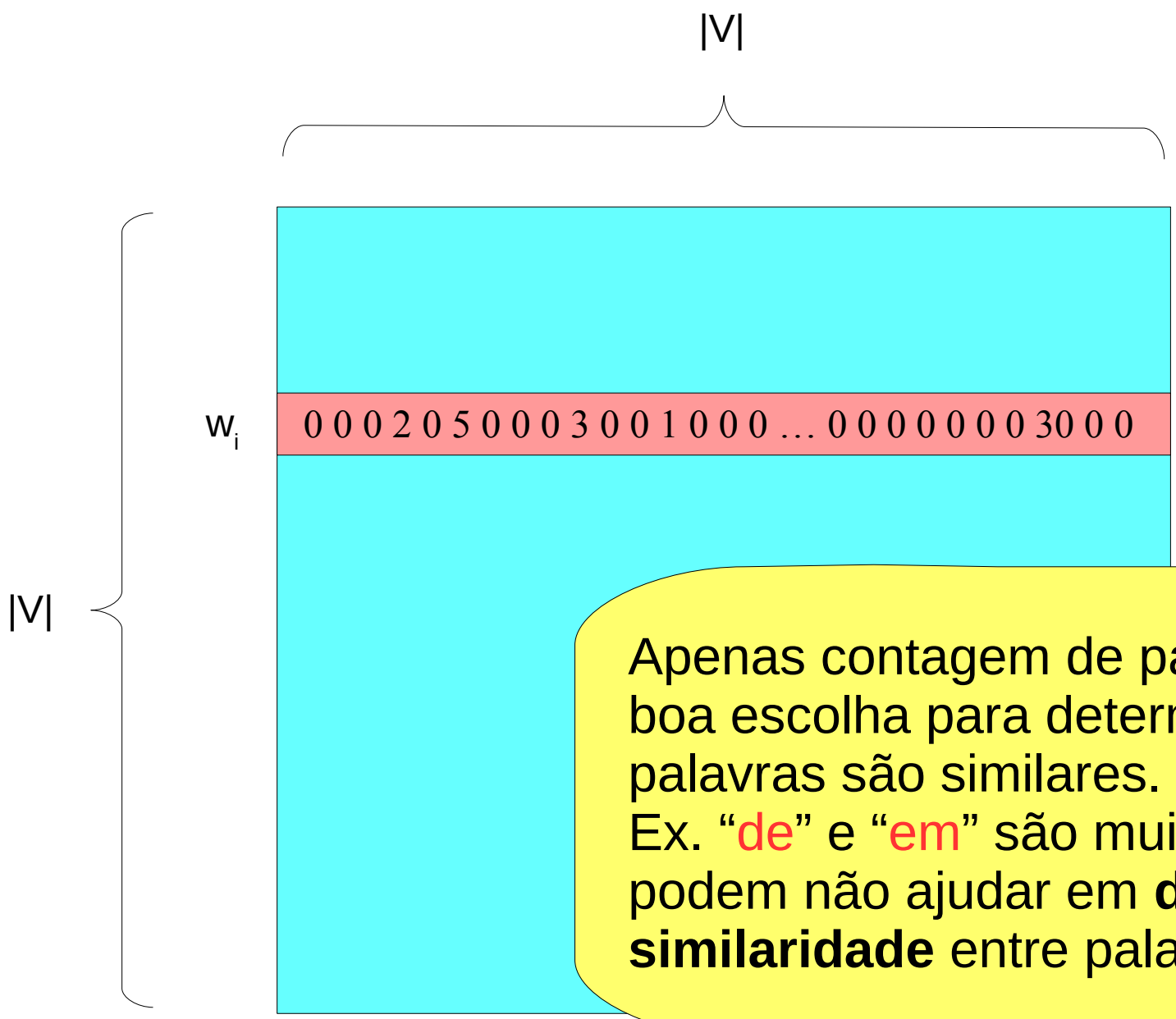
O significado de uma palavra é calculada **a partir da distribuição de palavras** que **ao redor dela**.

- Em tópicos anteriores uma palavra era representada por **um índice** em um vocabulário.
- Nesta abordagem: as palavras são representadas como um **vetor de números**. (*embedding into a vector*)

Matriz termo-contexto



Matriz termo-contexto



Apenas contagem de palavras não é boa escolha para determinar se 2 palavras são similares.
Ex. “de” e “em” são muito frequentes, e podem não ajudar em **discriminar similaridade** entre palavras.

Pointwise Mutual Information (PMI)
Positive Pointwise Mutual Information (PPMI)

Ambas medidas permitem mensurar quão **informativa** é a palavra de contexto, dada uma palavra (alvo)

Similaridade por acaso?

As palavras:

“de”, “em”, “para”, “quem”, “está”, “eles”, “nossos”…

Não são informativas pois estão presentes em quase todos os contextos.

A melhor ponderação ou medida de associação entre palavras deve nos dizer com **que frequência mais do que o acaso as duas palavras co-ocorrem**.

Similaridade por acaso?

Em um contexto de pesquisa científica, as palavras:

“método”, “proposta”, “metodologia”, “análise”, “estudo”…

Não são informativas pois estão presentes em quase todos os contextos de pesquisa.

Pointwise Mutual Information (PMI)

É uma medida baseada na medida de **Informação Mútua**:

$$I(X,Y) = \sum_x \sum_y P(x,y) \log_2 \frac{P(x,y)}{P(x)P(y)}$$

Informação Mútua:

- é uma medida que representa a quantidade de informação que uma variável aleatória contém sobre outra variável aleatória.
- é uma medida de dependência entre variáveis aleatórias.
- é uma medida da sobreposição de informações entre duas variáveis aleatórias.

Pointwise Mutual Information (PMI)

PMI é uma medida de associação que mede quão frequente dois eventos (x e y) ocorrem se ambos são independentes:

$$I(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

A razão entre a prob. Conjunta dos eventos e suas prob. separadas

Para palavras mede o quanto 2 palavras estão associadas.

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

$$\mathrm{PMI}(word_1, word_2) = \log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}$$

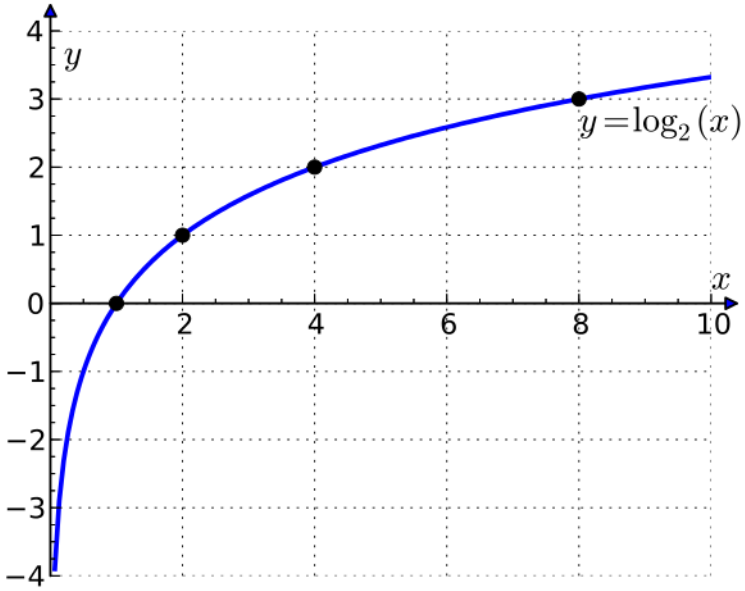
Pointwise Mutual Information (PMI)

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

Se <1 o PMI será negativo

PMI terá valores entre [-∞, +∞]:

- **Negativos**: implica que as coisas estão ocorrendo com menos frequência do que esperávamos por acaso.

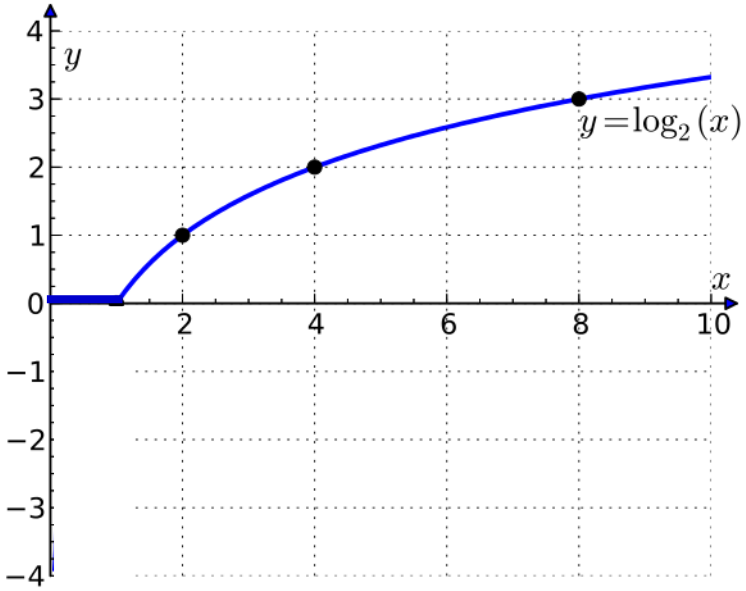


Valores não confiáveis para corpus pequeno!

Pointwise Mutual Information
Positive Pointwise Mutual Information

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

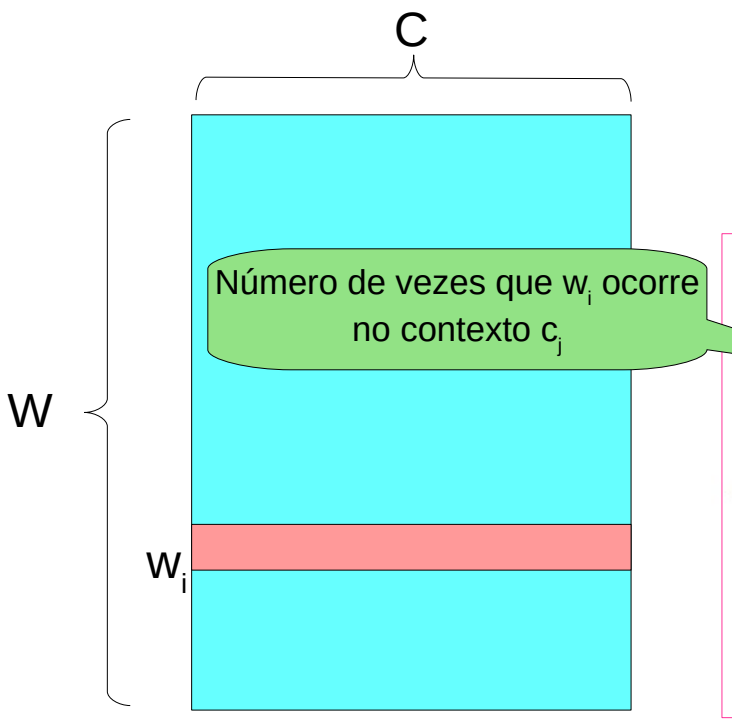
$$\text{PPMI}(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$



O mais comum é substituir os valores negativos do PMI por zero.

Calculando PPMI em uma matriz termo-contexto

Seja F a matriz com W linhas (palavras) e C colunas (contextos)



$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}}$$

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

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

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

$$ppmi_{ij} = \begin{cases} pmi_{ij} & \text{if } pmi_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Atividade 1

Calcule:

`PPMI(w=information, c=data)`

da seguinte matriz termo-contexto

	computer	data	pinch	result	sugar
apricot	0	0	1	0	1
pineapple	0	0	1	0	1
digital	2	1	0	1	0
information	1	6	0	4	0

Atividade 1

Calcule:

`PPMI(w=information, c=data)`

da seguinte matriz termo-contexto

	computer	data	pinch	result	sugar
apricot	0	0	1	0	1
pineapple	0	0	1	0	1
digital	2	1	0	1	0
information	1	6	0	4	0

```
P(w=information, c=data) =  6/19 = 0.32
P(w=information)         = 11/19 = 0.58
P(c=data)                =  7/19 = 0.37
```

1.49

```
PPMI(w=information, c=data) = log2(0.32/(0.58*0.37)) = 0.5764
```

	computer	data	pinch	result	sugar
apricot	0	0	1	0	1
pineapple	0	0	1	0	1
digital	2	1	0	1	0
information	1	6	0	4	0

	p(w,context)					p(w)
	computer	data	pinch	result	sugar	
apricot	0.00	0.00	0.05	0.00	0.05	0.11
pineapple	0.00	0.00	0.05	0.00	0.05	0.11
digital	0.11	0.05	0.00	0.05	0.00	0.21
information	0.05	0.32	0.00	0.21	0.00	0.58
p(context)	0.16	0.37	0.11	0.26	0.11	

-0.63

PPMI(w,context)

	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

PPMI(w,context)

	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

O PPMI tem valores altos para eventos pouco frequentes (palavras muito raras)

Uma maneira de reduzir esse viés é considerar:

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

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

Levy, O., Goldberg, Y., & Dagan, I. (2015). Improving distributional similarity with lessons learned from word embeddings. Transactions of the Association for Computational Linguistics, 3, 211-225.

Improving Distributional Similarity with Lessons Learned from Word Embeddings

Omer Levy **Yoav Goldberg** **Ido Dagan**
Computer Science Department
Bar-Ilan University
Ramat-Gan, Israel
{omerlevy,yogo,dagan}@cs.biu.ac.il

Abstract

Recent trends suggest that neural-network-inspired word embedding models outperform traditional count-based distributional models on word similarity and analogy detection tasks. We reveal that much of the performance gains of word embeddings are due to certain system design choices and hyperparameter optimizations, rather than the embedding algorithms themselves. Furthermore, we show that these modifications can be transferred to traditional distributional models, yielding similar gains. In contrast to prior reports, we observe mostly local or insignificant performance differences between the methods, with no global advantage to any single approach over the others.

A recent study by Baroni et al. (2014) conducts a set of systematic experiments comparing `word2vec` embeddings to the more traditional distributional methods, such as pointwise mutual information (PMI) matrices (see Turney and Pantel (2010) and Baroni and Lenci (2010) for comprehensive surveys). These results suggest that the new embedding methods consistently outperform the traditional methods by a non-trivial margin on many similarity-oriented tasks. However, state-of-the-art embedding methods are all based on the same bag-of-contexts representation of words. Furthermore, analysis by Levy and Goldberg (2014c) shows that `word2vec`'s SGNS is implicitly factorizing a word-context PMI matrix. That is, the mathematical objective and the sources of information available to SGNS are in fact very similar to those employed by the more traditional methods.

What, then, is the source of superiority (or per-

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

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

$$\alpha = 0.75$$

Quando c é raro, esta expressão permite diminuir o PPMI

Uma alternativa é usar uma estratégia que tenha um **efeito similar**:

Suavisação:

- Add-**k** smoothing
- Laplace smoothing

Add-2 Smoothed Count(w,context)

	computer	data	pinch	result	sugar
apricot	2	2	3	2	3
pineapple	2	2	3	2	3
digital	4	3	2	3	2
information	3	8	2	6	2

p(w,context) [add-2] **p(w)**

	computer	data	pinch	result	sugar	
apricot	0.03	0.03	0.05	0.03	0.05	0.20
pineapple	0.03	0.03	0.05	0.03	0.05	0.20
digital	0.07	0.05	0.03	0.05	0.03	0.24
information	0.05	0.14	0.03	0.10	0.03	0.36
p(context)	0.19	0.25	0.17	0.22	0.17	

PPMI(w,context)

	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

PPMI(w,context) [add-2]

	computer	data	pinch	result	sugar
apricot	0.00	0.00	0.56	0.00	0.56
pineapple	0.00	0.00	0.56	0.00	0.56
digital	0.62	0.00	0.00	0.00	0.00
information	0.00	0.58	0.00	0.37	0.00

test3.py

```
# contabilizando os pares de palavras
k           = 3
Mcontext    = numpy.ones((V, V))*2  # add-2 smoothing
Vocabulary  = list(Vocabulary)
iVocabulary = dict([])

for (i,w) in enumerate(Vocabulary):
    iVocabulary[w] = i

for d in Document.keys():
    print (d)
    for (i,w) in enumerate(Document[d]):
        context = []
        if i>k:
            context += Document[d][i-k:i]
        if i<len(Document[d])-k:
            context += Document[d][i+1:i+k+1]

        print (i, w, context)

        iw = iVocabulary[w]
        for wc in context:
            Mcontext[iw, iVocabulary[wc]] += 1
```

test3.py

```
# Calculando para cada palavra do vocabulario: PPMI
N    = numpy.sum(Mcontext)
PPMI = numpy.zeros((V, V))

Fw   = numpy.sum(Mcontext, axis=1) # somatoria de cada linha
Fc   = numpy.sum(Mcontext, axis=0) # somatoria de cada coluna
```

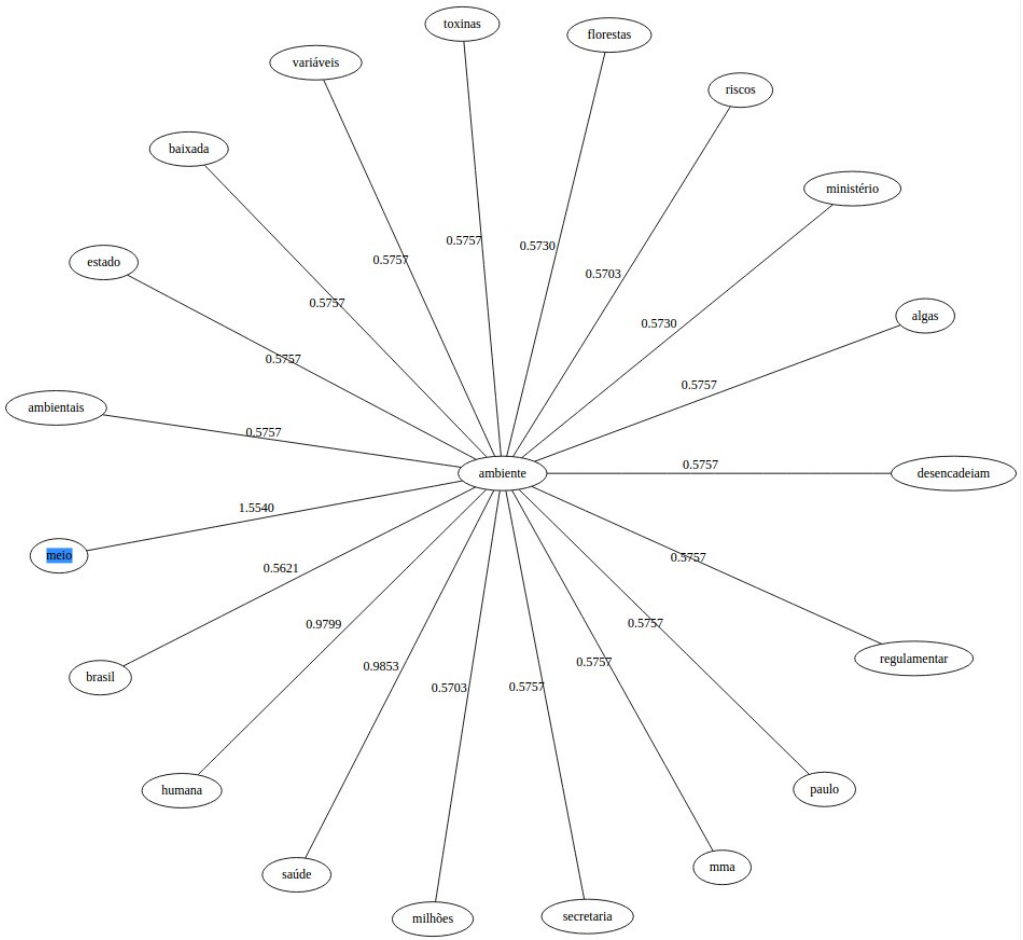
```
for i in range(0, V):
    for j in range(0, V):
        PPMI[i,j] = max(0, math.log2((Mcontext[i,j]/N)/(Fw[i]/N*Fc[j]/N)) )
```

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}}$$

$$ppmi_{ij} = \begin{cases} pmi_{ij} & \text{if } pmi_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

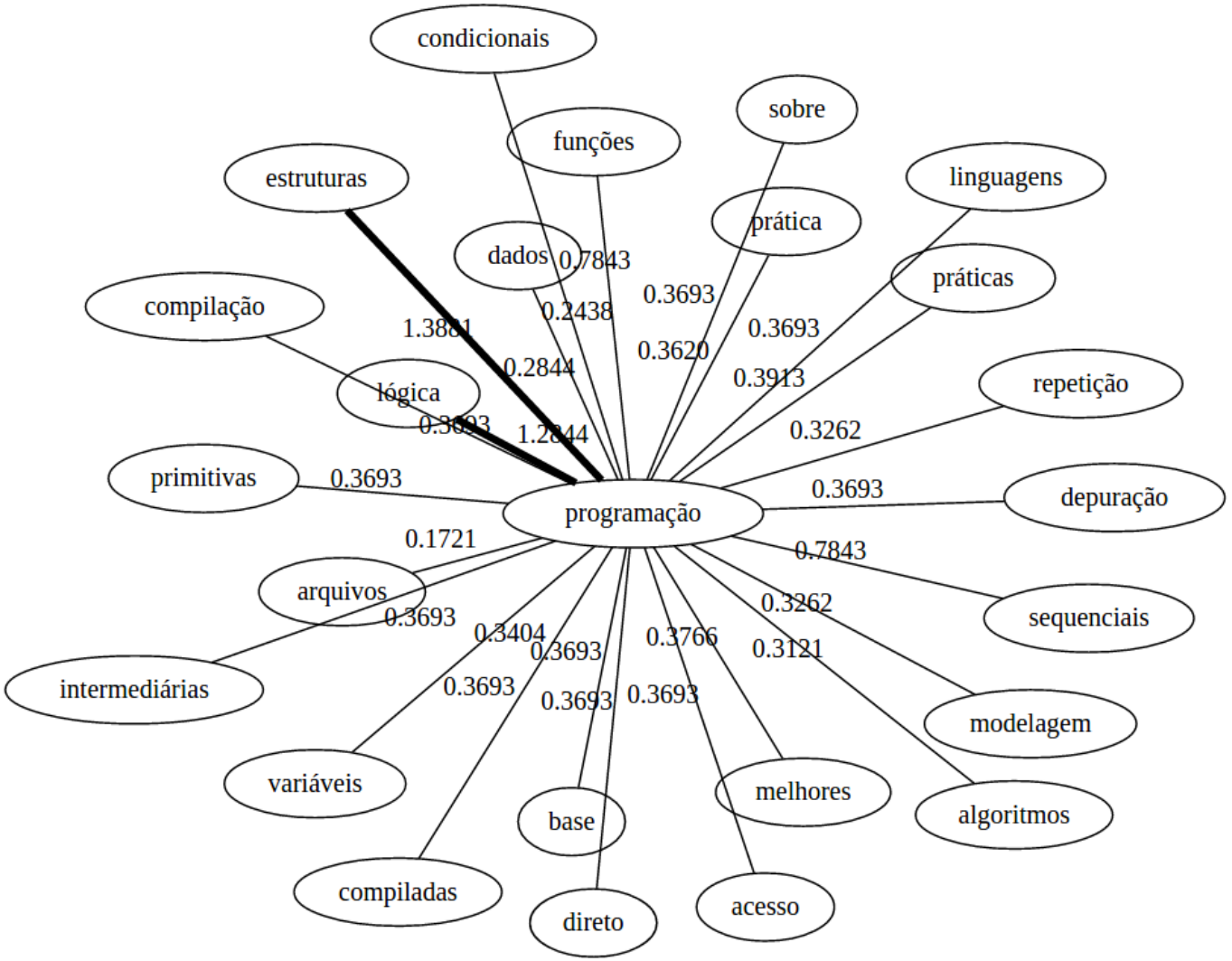
test3.py

```
python3 teste3.py noticias
```

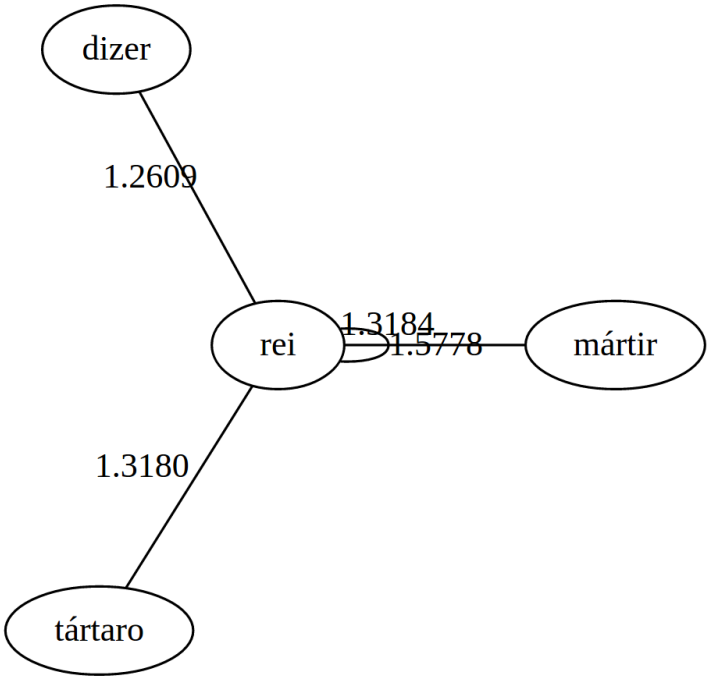


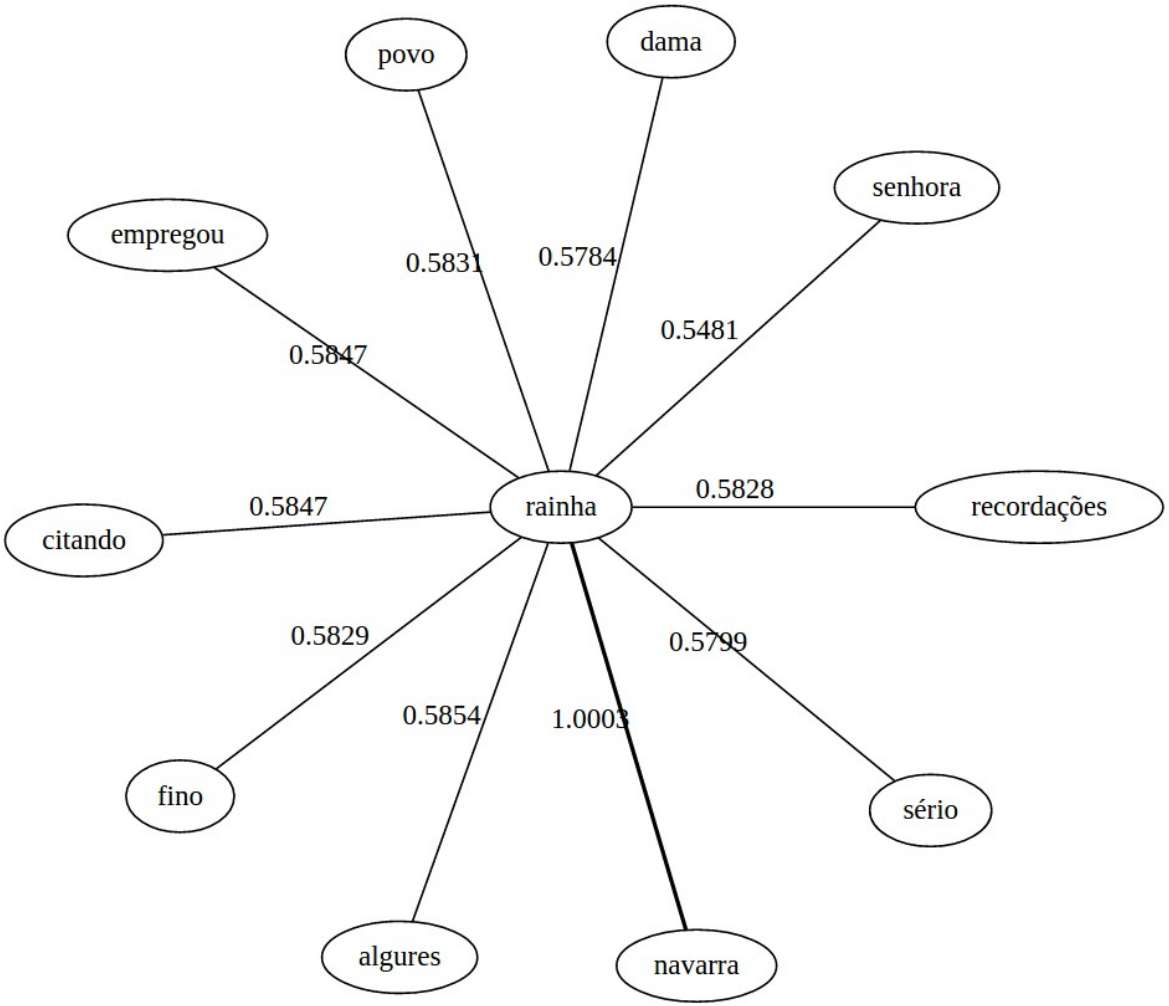
test3.py

```
python3 teste3.py ufabc-bcc
```

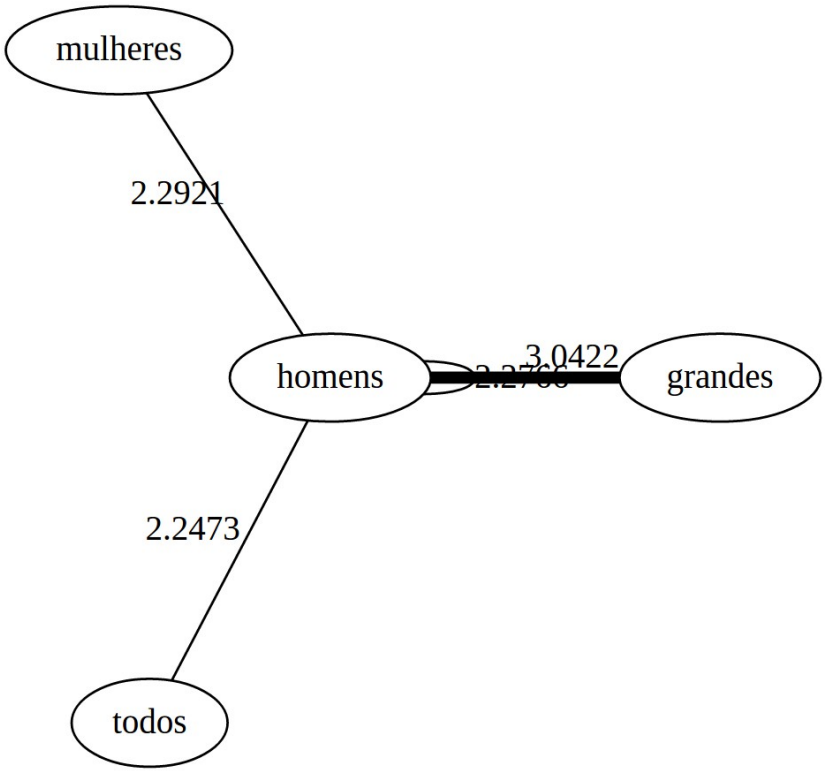


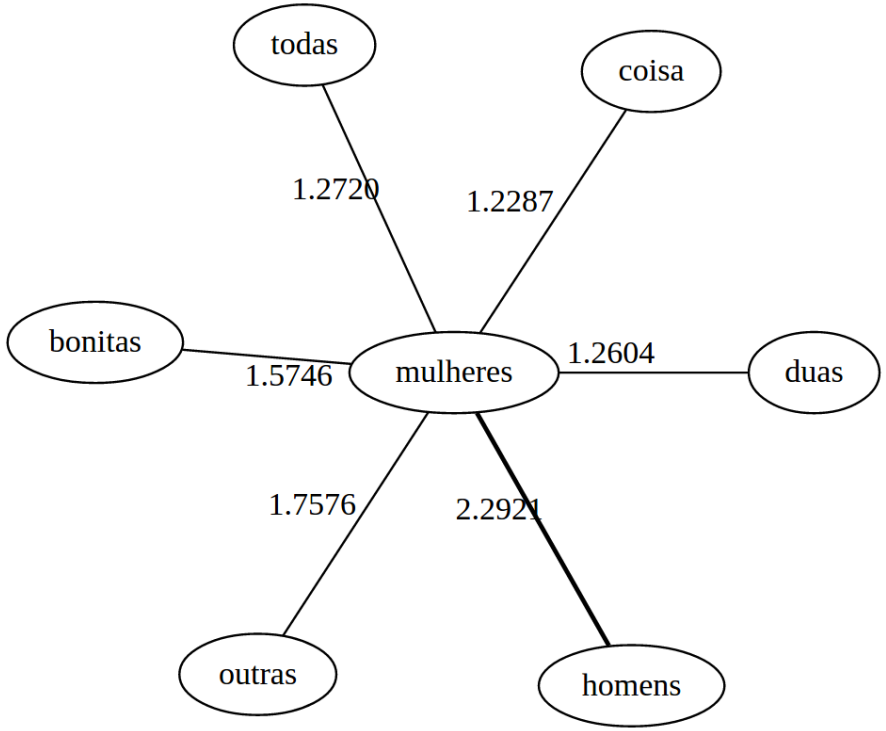
machado-db


dizer
1.2609
rei
1.3184
1.5778
mártir
1.3180
tártaro


povo
dama
senhora
empregou
0.5831
0.5784
0.5481
0.5847
0.5847
rainha
0.5828
recordações
citando
0.5829
0.5799
0.5854
1.0003
fino
sério
algures
navarra

machado-db


mulheres
2.2921
homens
3.0422
grandes
2.2473
todos


todas
coisa
1.2720
1.2287
bonitas
1.5746
mulheres
1.2604
duas
1.7576
2.2921
outras
homens

Outras medidas de similaridade?

$$\text{PMI}(w,f) = \log_2 \frac{P(w,f)}{P(w)P(f)}$$

$$\text{t-test}(w,f) = \frac{P(w,f)-P(w)P(f)}{\sqrt{P(f)P(w)}}$$

$$\text{cosine}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

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

$$\text{Dice}(\vec{v},\vec{w}) = \frac{2\times\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} (v_i+w_i)}$$

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

Desafio 2: Bônus +0.5 na MF

Resumo de uma publicação descrita na lista
https://en.wikipedia.org/wiki/List_of_important_publications_in_computer_science

Resumo nos mesmos moldes dos resumos de aula.

Porque o trabalho é considerado importante para a área de computação?

Envio pelo tidia (seção atividades).

Desafio 2 (para o final de quadri)

WIKIPEDIA
The Free Encyclopedia

Not logged in Talk Contributions Create account Log i

Article Talk Read Edit View history Search Wikipedia

List of important publications in computer science

From Wikipedia, the free encyclopedia

Contents [hide]

1 Artificial intelligence
2 Collaborative networks
3 Compilers
4 Computer architecture
5 Computer graphics
6 Computer vision
7 Concurrent, parallel, and distributed computing
8 Databases
9 History of computation
10 Information retrieval
11 Networking
12 Operating systems
13 Programming languages
14 Scientific computing
15 Software engineering
16 Security
17 Theoretical computer science
18 See also
19 References
20 External links

Desafio 2 (para o final de quadri)

Information retrieval [edit]

Sugestão de artigos

A Vector Space Model for Automatic Indexing [edit]

- Gerard Salton, A. Wong, C. S. Yang
- Commun. ACM 18(11): 613–620 (1975)

Description: Presented the vector space model.

Extended Boolean Information Retrieval [edit]

- Gerard Salton, Edward A. Fox, Harry Wu
- Commun. ACM 26(11): 1022–1036 (1983)

Description: Presented the inverted index

A Statistical Interpretation of Term Specificity and Its Application in Retrieval [edit]

- Karen Spärck Jones
- Journal of Documentation 28: 11–21 (1972). doi:10.1108/eb026526.

Description: Conceived a statistical interpretation of term specificity called Inverse document frequency (IDF), which became a cornerstone of term weighting.