MCZA017-13
Processamento de Linguagem Natural

Semântica e similaridade de palavras: Parte II

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

2Q-2019

Matriz termo-documento

Matriz de co-ocorrência: termo-documento

Dois **documentos** são similares se os vetores são similares

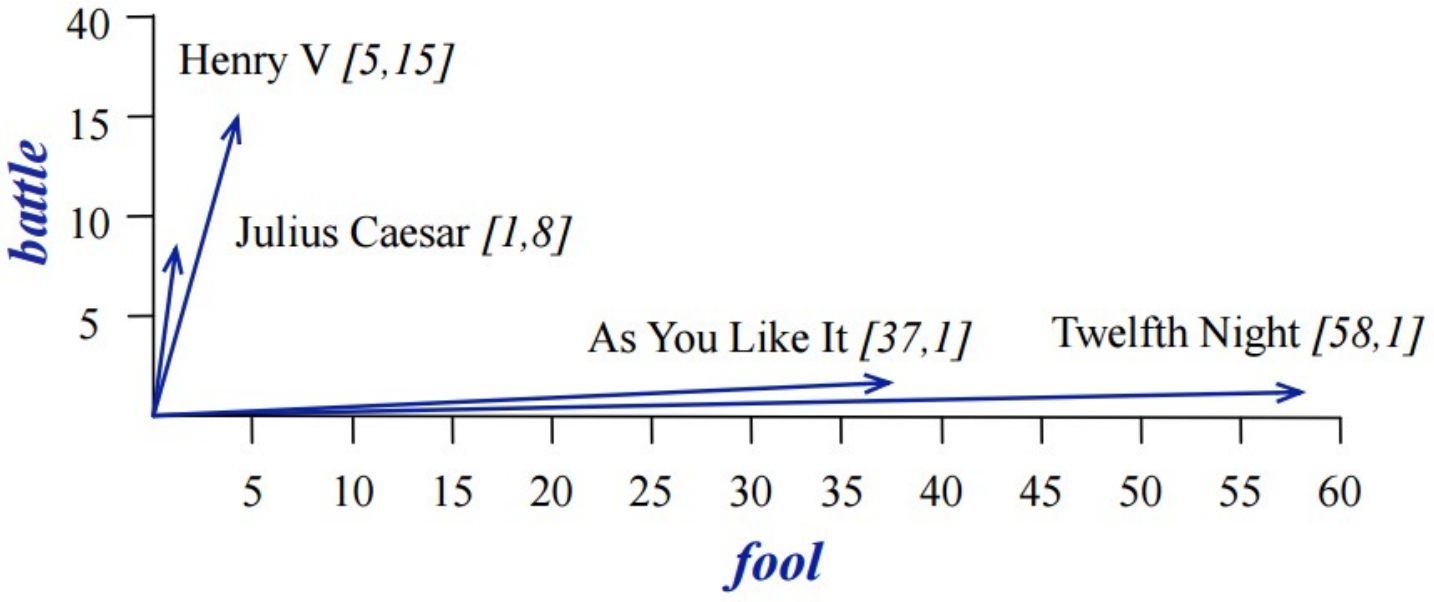
V

	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

	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 de co-ocorrência: termo-documento

Duas **palavras** são similares 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

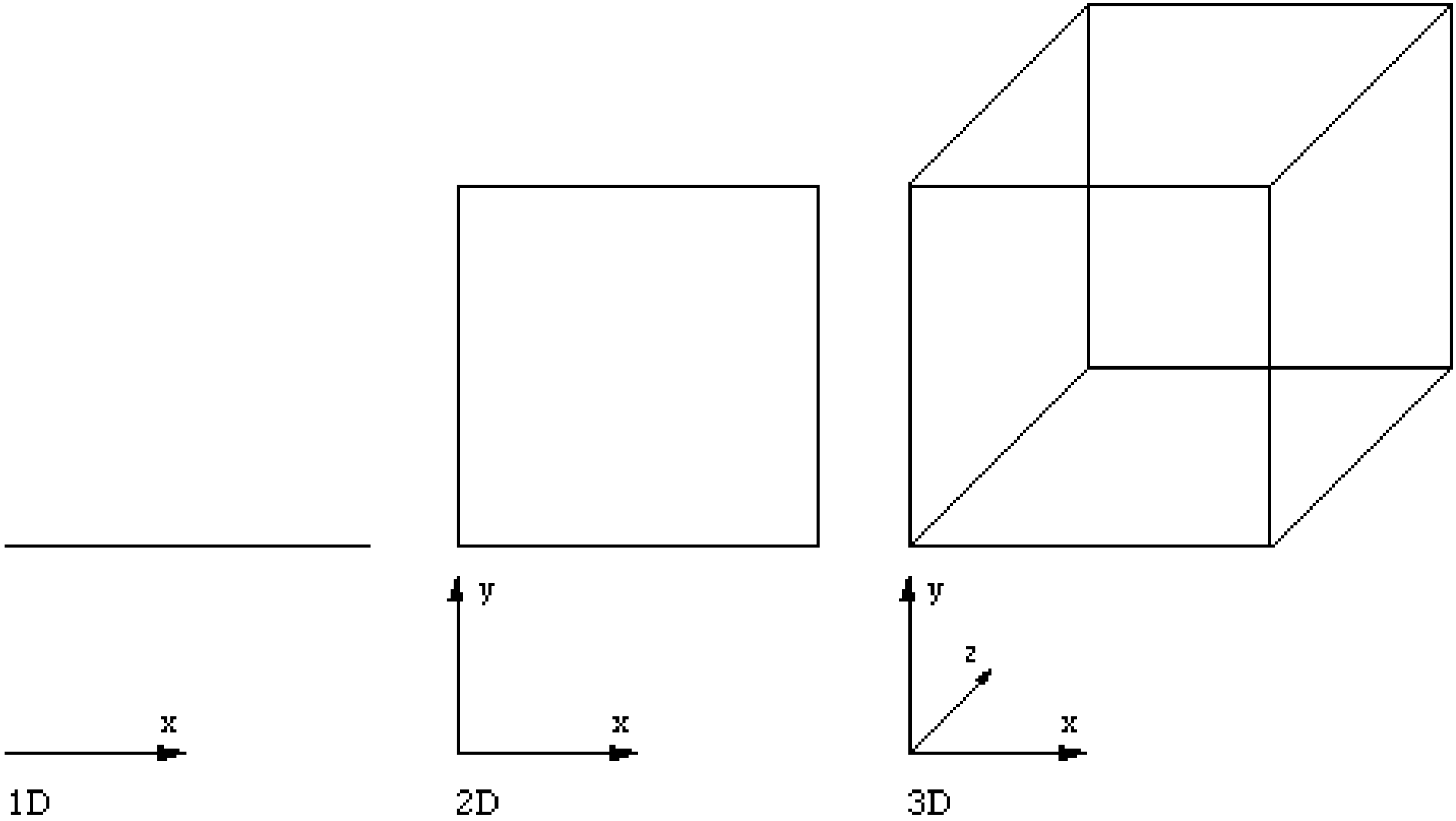
A dimensão do vetor é o número de documentos: $N^{|D|}$

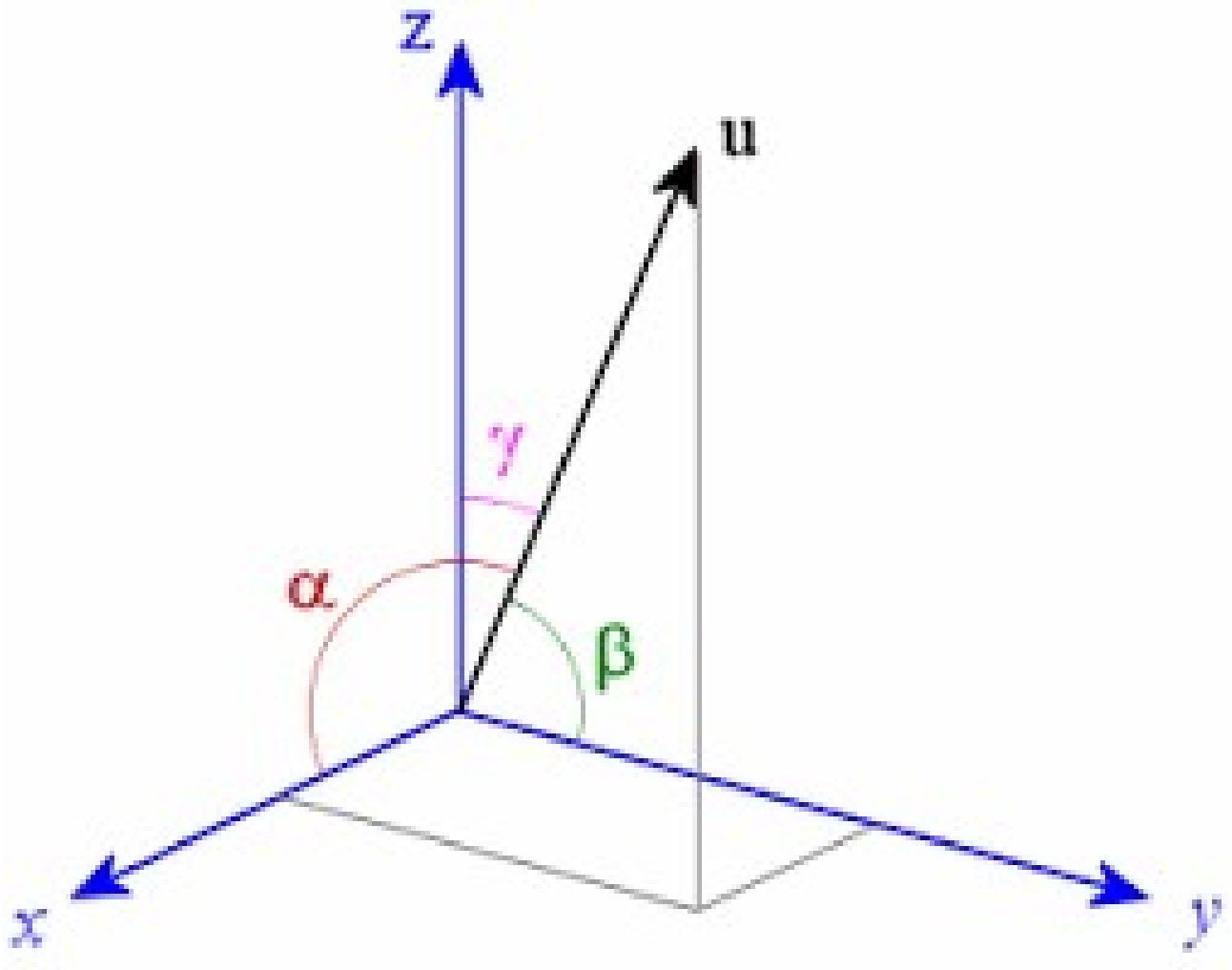
Matriz de co-ocorrência: termo-documento

- 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.

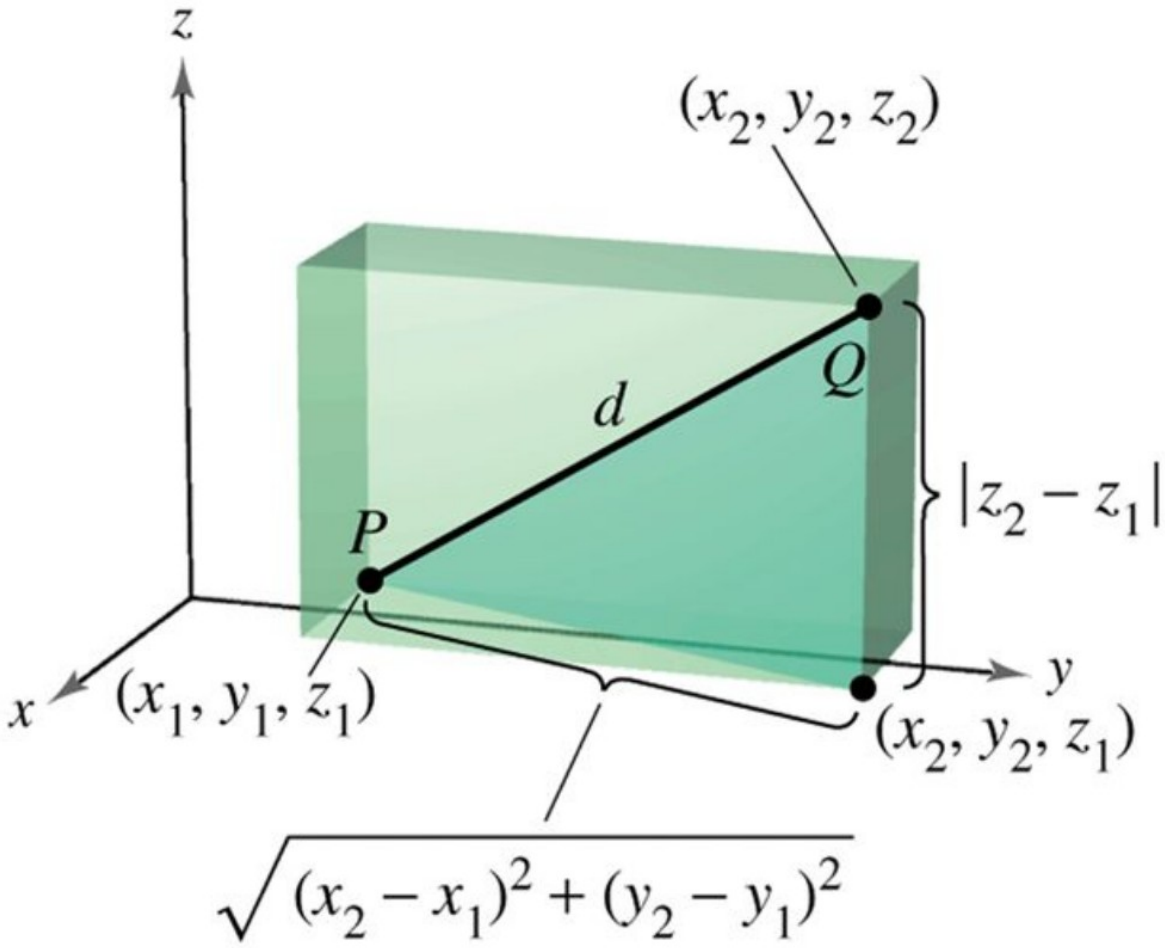
	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

Medidas de similaridade

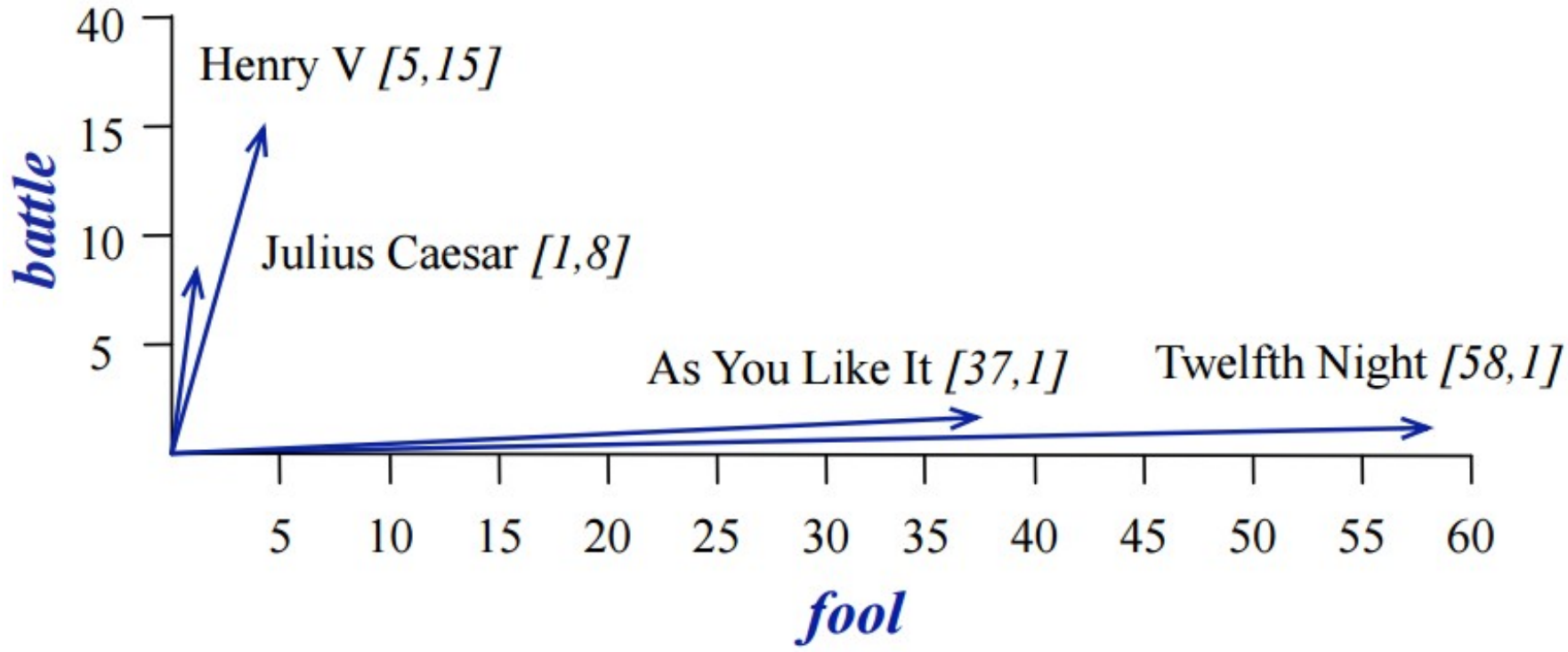

x
y
x
y
z
x
1D
2D
3D


z
u
γ
α
β
x
y

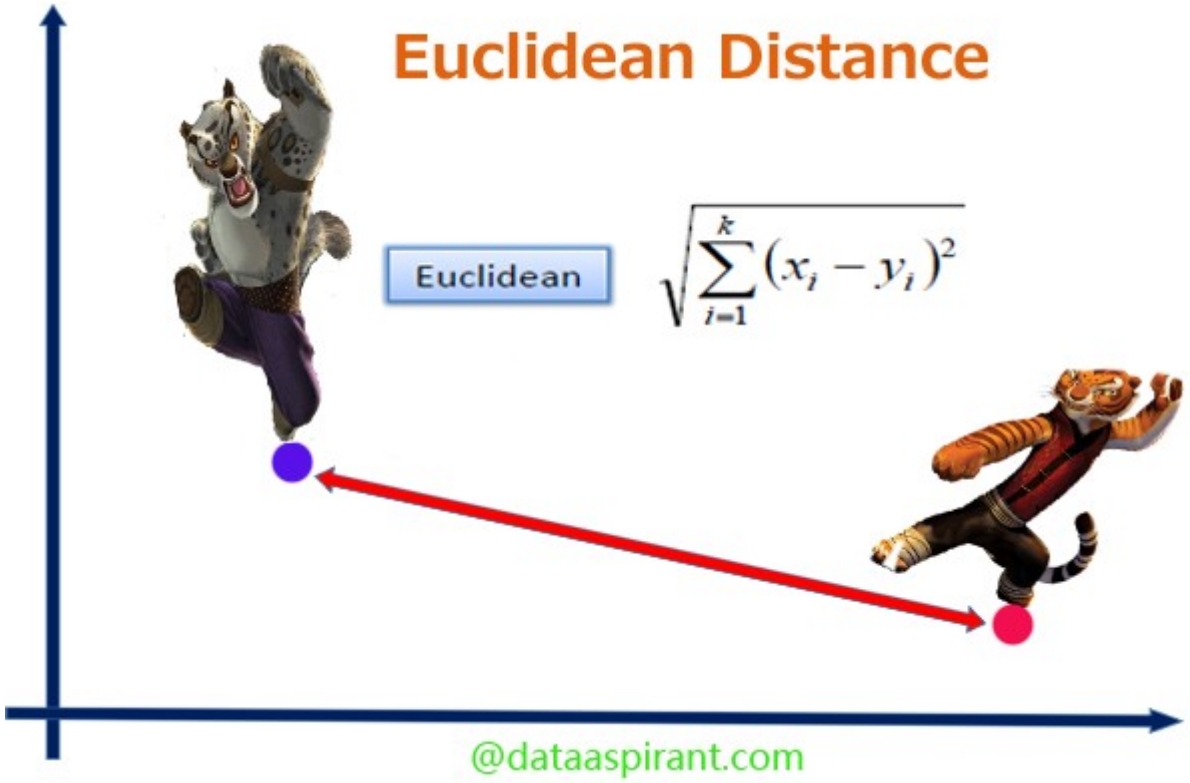
Length: $\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + v_3^2}$ $\quad d = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2 + (z_2-z_1)^2}$



Vetores 2D dos quatro documentos

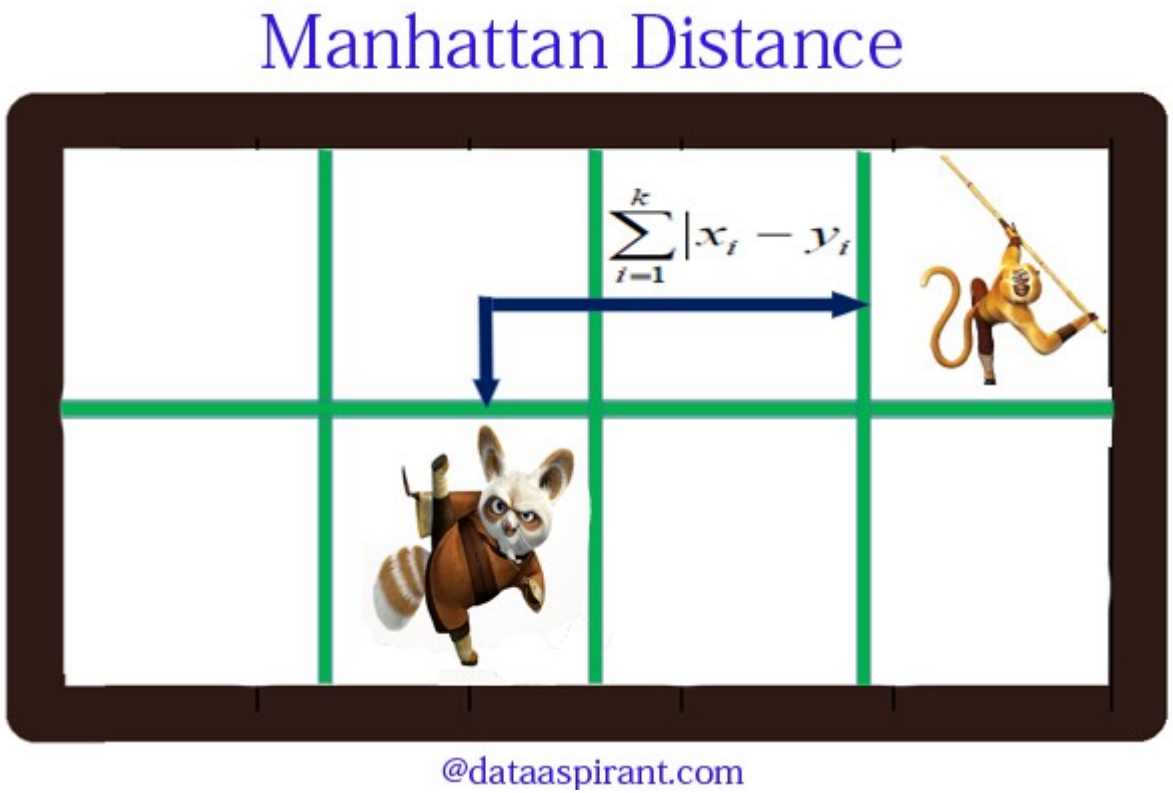


Medidas de similaridade



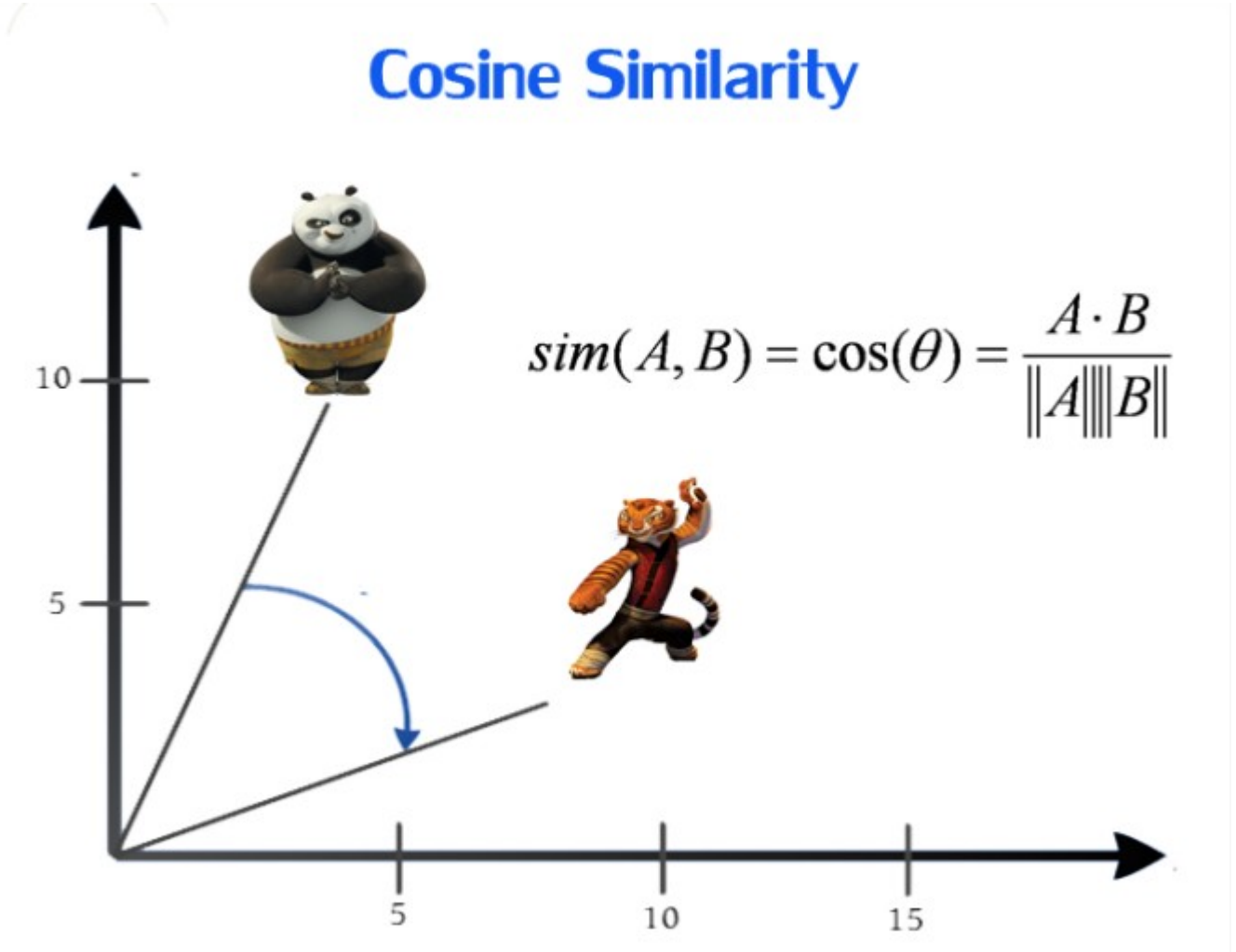
https://dataconomy.com/2015/04/implementing-the-five-most-popular-similarity-measures-in-python/

Medidas de similaridade



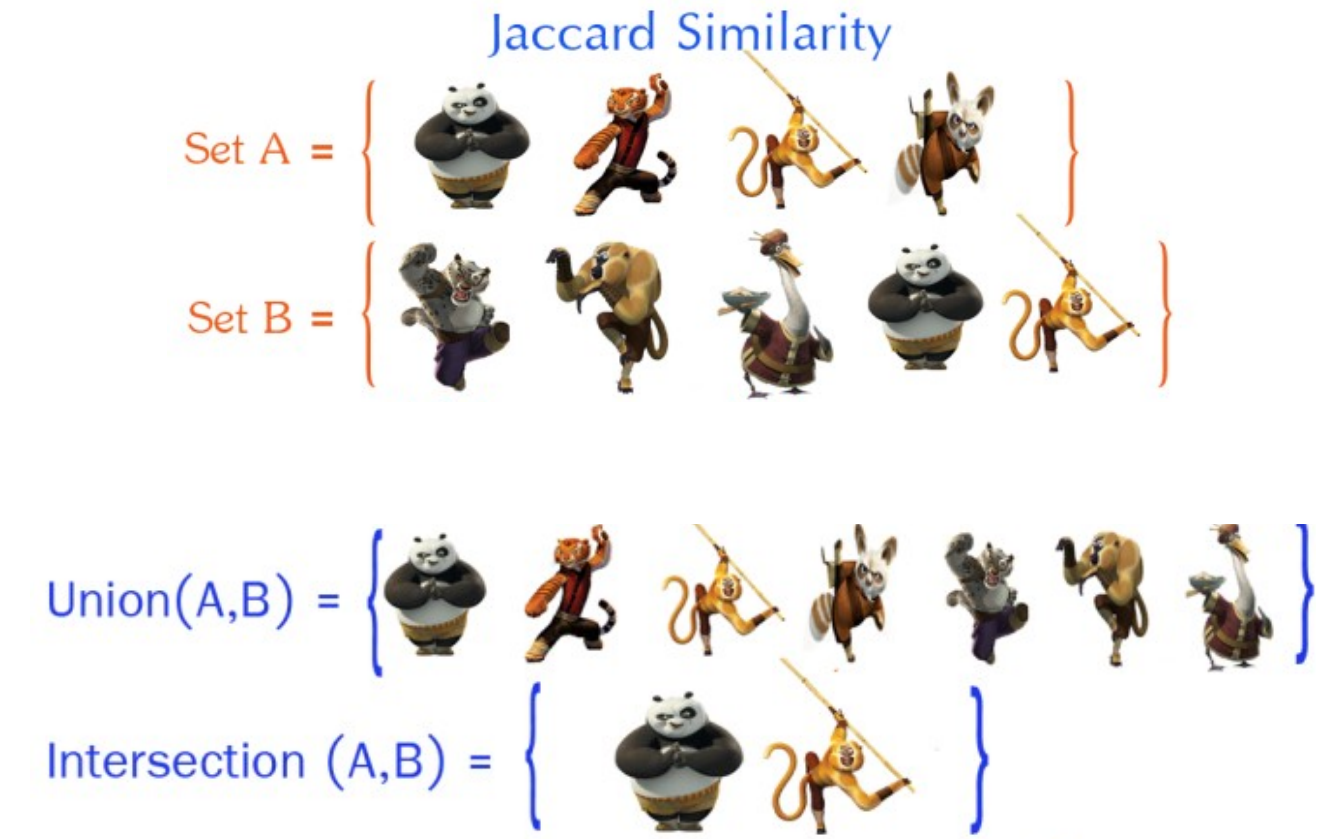
https://dataconomy.com/2015/04/implementing-the-five-most-popular-similarity-measures-in-python/

Medidas de similaridade



https://dataconomy.com/2015/04/implementing-the-five-most-popular-similarity-measures-in-python/

Medidas de similaridade



https://dataconomy.com/2015/04/implementing-the-five-most-popular-similarity-measures-in-python/

Medidas de similaridade

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

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

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

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

Similaridade entre documentos

Matriz de co-ocorrência: termo-documento

Dois **documentos** são similares 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: $N^{|V|}$

teste1.py: similaridade entre documentos

```
if __name__ == '__main__':
    dirDB = sys.argv[1]

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


    # leitura dos documentos
    for fileName in os.listdir(dirDB):
        document   = open(dirDB+"/"+fileName,'r')
        content    = document.read()
        words      = re.findall(regex, content)
        Document[fileName] = words
        Vocabulary.update(words)
    D = len(Document)
    V = len(Vocabulary)
    print("Numero de documentos  : {}".format( D ))
    print("Tamanho do vocabulario: {}".format( V ))
```

teste1.py: similaridade entre documentos

```
# calculando as frequencias das palavras nas obras
M          = numpy.zeros((V, D))
documents  = list(Document.keys())
vocabulary = list(Vocabulary)
for j in range(0, D):
    d = documents[j]
    print (d)
    for i in range(0, V):
        w      = vocabulary[i]
        M[i,j] = Document[d].count(w)


# distancia entre documentos
dist = numpy.ones((D,D))*numpy.nan
for d1 in range(0, D-1):
    for d2 in range(d1+1, D):
        dist[d1,d2] = numpy.linalg.norm(M[:,d1]-M[:,d2])
print(dist)
```

teste1.py: similaridade entre documentos

```
# criando o grafo de documentos (similaridade entre documentos)
dist = 1 - (dist-numpy.nanmin(dist))/(numpy.nanmax(dist)-numpy.nanmin(dist))

txtGraph = "\ngraph{"
for d1 in range(0, D-1):
    for d2 in range(d1+1, D):
        if dist[d1,d2]!=numpy.nan:
            txtGraph += '\n "{0}" -- "{1}"[label="{2:.2f}", penwidth={2:.2f}]'.
txtGraph += "\n}"

print(txtGraph)
```

ufabc-bcc

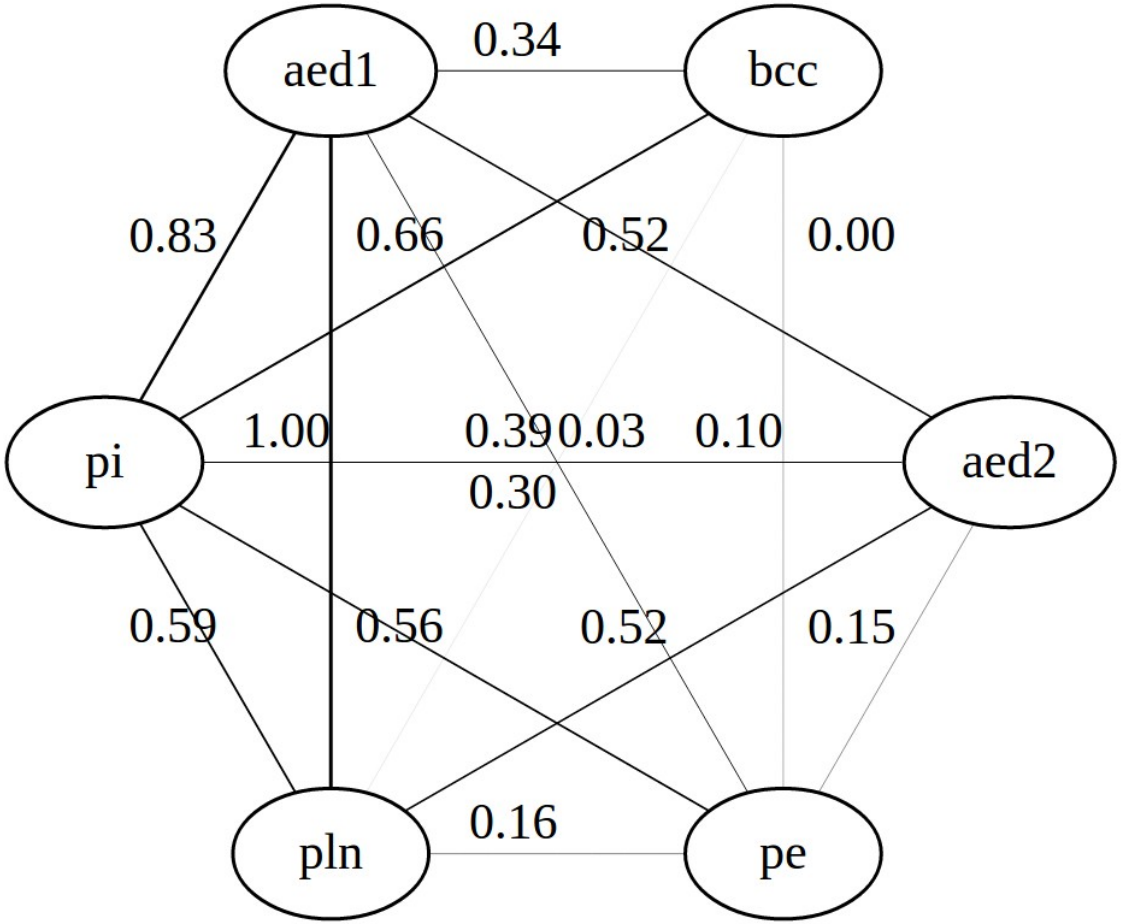
aed1	breve introdução à linguagem c. noções básicas de análise de complexidade de tempo de algoritmos. estruturas lineares: busca e ordenação. árvores de busca. árvores balanceadas.
aed2	hashing. introdução a arquivos. arquivos seqüenciais. arquivos indexados. arquivos de acesso direto. prática de programação dos arquivos e das funções primitivas na resolução de problemas. compressão de arquivos.
bcc	fundamentos da computação; representação gráfica de funções; noções de estatística, correlação e regressão; base de dados; lógica de programação: variáveis e estruturas sequenciais; lógica de programação: estruturas condicionais; lógica de programação: estruturas de repetição; modelagem e simulação computacional: conceitos fundamentais; modelagem e simulação computacional: a ciência na prática.
pe	apresentar noções básicas e intermediárias sobre algoritmos, programação em linguagens compiladas, compilação, programas em execução (processos), ponteiros, alocação estática e dinâmica de memória, vetores e matrizes, funções e passagem de parâmetros, registros, arquivos e recursividade. aplicar todos os conceitos apresentados no contexto da resolução de problemas clássicos e novos da computação.
pi	introdução a algoritmos. variáveis e tipos de dados. operadores aritméticos, lógicos e precedência. métodos/funções e parâmetros. estruturas de seleção. estruturas de repetição. vetores. matrizes. entrada e saída de dados. depuração. melhores práticas de programação.
pln	introdução a algoritmos. variáveis e tipos de dados. operadores aritméticos, lógicos e precedência. métodos/funções e parâmetros. estruturas de seleção. estruturas de repetição. vetores. matrizes. entrada e saída de dados. depuração. melhores práticas de programação.

teste1.py: ufabc-bcc

```
python3 teste1.py ufabc-bcc/
Numero de documentos  : 6
Tamanho do vocabulario: 109

[[nan  10.24695077   8.48528137   9.59166305  10.44030651   10.19803903]
 [nan          nan   9.21954446   8.66025404  10.86278049    8.66025404]
 [nan          nan          nan   7.34846923   8.06225775    8.36660027]
 [nan          nan          nan          nan   9.43398113    6.63324958]
 [nan          nan          nan          nan          nan   10.72380529]
 [nan          nan          nan          nan          nan           nan]]

graph{
 "pe" -- "aed2"[label="0.15", penwidth=0.15]
 "pe" -- "pi"  [label="0.56", penwidth=0.56]
 "pe" -- "aed1"[label="0.30", penwidth=0.30]
 "pe" -- "bcc" [label="0.10", penwidth=0.10]
 "pe" -- "pln" [label="0.16", penwidth=0.16]
...
 "aed1" -- "bcc"[label="0.34", penwidth=0.34]
 "aed1" -- "pln"[label="1.00", penwidth=1.00]
 "bcc" -- "pln" [label="0.03", penwidth=0.03]
}
```

ufabc-bcc

https://dreampuf.github.io/GraphvizOnline/

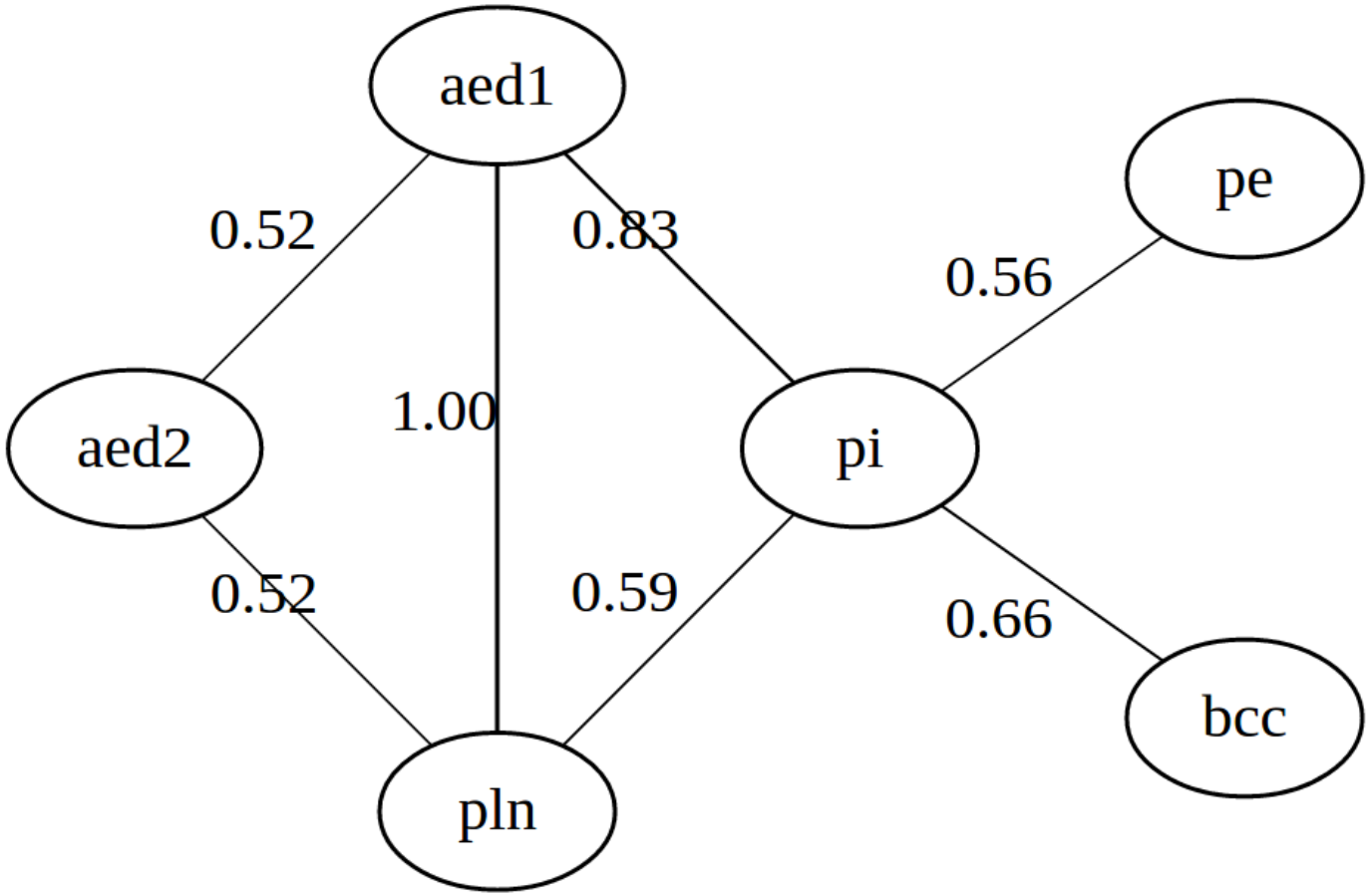


ufabc-bcc

```
txtGraph = "\ngraph{"
for d1 in range(0, D-1):
    for d2 in range(d1+1, D):
        if dist[d1,d2]!=numpy.nan and dist[d1,d2]>=0.5:
            txtGraph += '\n "{0}" -- "{1}"[label="{2:.2f}", penwidth={2:.2f}]'
txtGraph += "\n}"

print(txtGraph)
```

ufabc-bcc

https://dreampuf.github.io/GraphvizOnline/



machado-db

Dom Casmurro
Esaú e Jacó
Memórias Póstumas de Brás Cubas
Quincas Borba

machado-db

CAPÍTULO PRIMEIRO

DO TÍTULO

Uma noite destas, vindo da cidade para o Engenho Novo, encontrei no trem da Central um rapaz aqui do bairro, que eu conheço de vista e de chapéu. Cumprimentou-me, sentou-se ao pé de mim, falou da Lua e dos ministros, e acabou recitando-me versos. A viagem era curta, e os versos pode ser que não fossem inteiramente maus. Sucedeu, porém, que, como eu estava cansado, fechei os olhos três ou quatro vezes; tanto bastou para que ele interrompesse a leitura e metesse os versos no bolso.

— Continue, disse eu acordando.

— Já acabei, murmurou ele.

— São muito bonitos.

Vi-lhe fazer um gesto para tirá-los outra vez do bolso, mas não passou do gesto; estava amuado. No dia seguinte entrou a dizer de mim nomes feios, e acabou alcunhando-me Dom Casmurro. Os vizinhos, que não gostam dos meus hábitos reclusos e calados, deram curso à alcunha, que afinal pegou. Nem

Dom Casmurro

Ao verme
que
primeiro roeu as frias carnes
do meu cadáver
dedico
como saudosa lembrança
estas
Memórias Póstumas

Prólogo da terceira edição

A primeira edição destas Memórias Póstumas de Brás Cubas foi feita aos pedaços na Revista Brasileira, pelos anos de 1880. Postas mais tarde em livro, corrigi o texto em vários lugares. Agora que tive de o rever para a terceira edição, emendei ainda alguma coisa e suprimi duas ou três dúzias de linhas. Assim composta, sai novamente à luz esta obra que alguma benevolência parece ter encontrado no público.

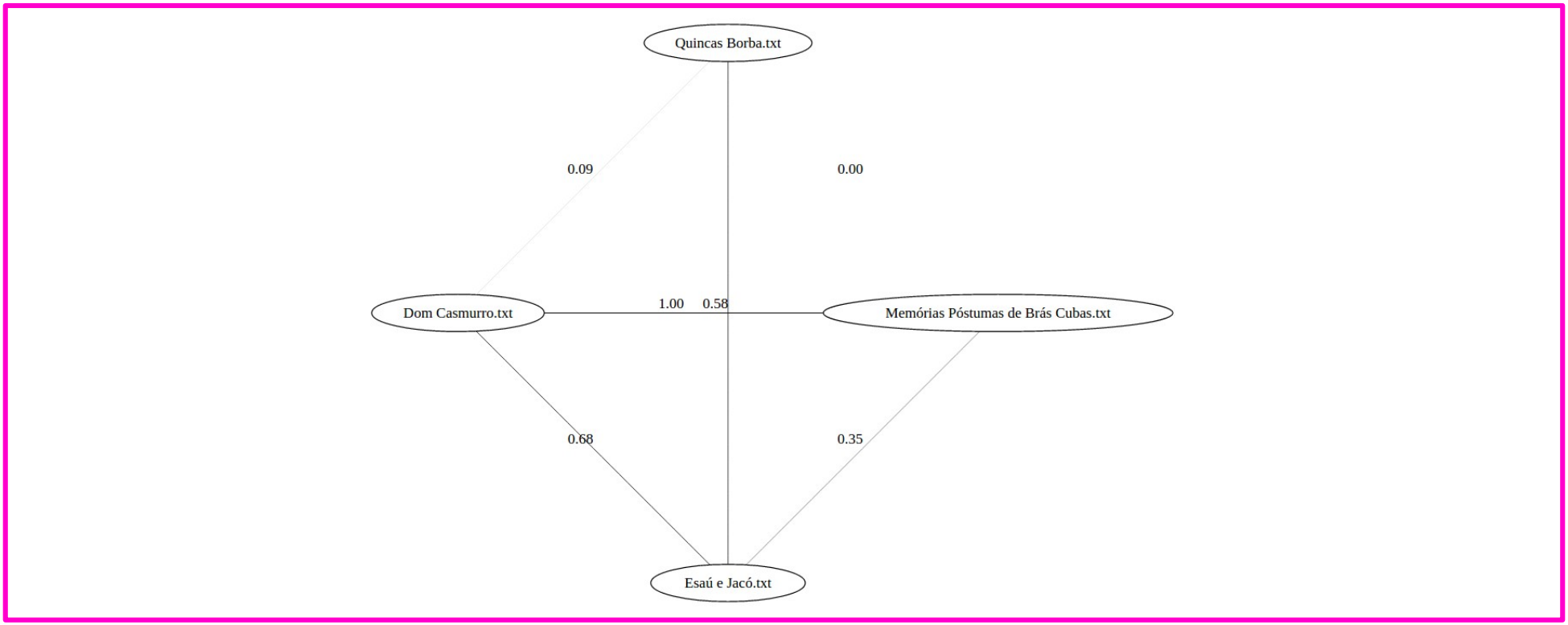
Capistrano de Abreu, noticiando a publicação do livro, perguntava: “As Memórias Póstumas de Brás Cubas são um romance?” Macedo Soares, em carta que me escreveu por esse tempo, recordava amigamente as Viagens na minha terra. Ao primeiro respondia já o defunto Brás Cubas (como o leitor viu e verá no prólogo dele que vai adiante) que sim e que não, que era romance

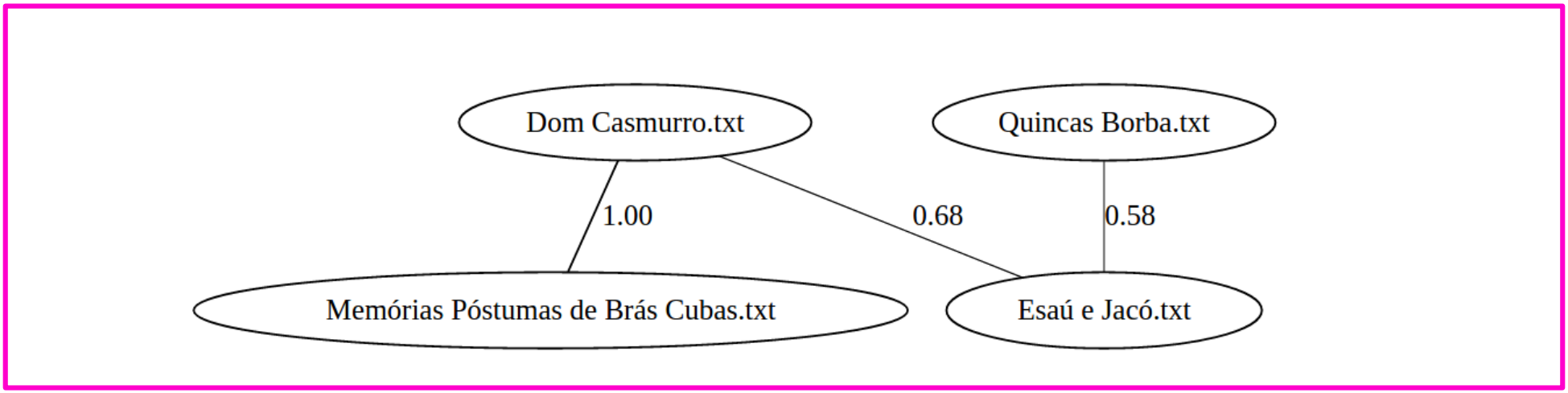
Memórias Póstumas de Brás Cubas

machado-db

```
python3 teste1.py machado-db

Numero de documentos  : 4
Tamanho do vocabulario: 25911

Dom Casmurro.txt
Memórias Póstumas de Brás Cubas.txt
Quincas Borba.txt
Esaú e Jacó.txt

[[          nan  1157.09982283  1640.27497695  1328.3075698 ]
 [          nan            nan  1686.16547231  1502.11018238]
 [          nan            nan            nan  1379.64886837]
 [          nan            nan            nan            nan]]

graph{
 ...
}
```

machado-db


Quincas Borba.txt
0.09
0.00
Dom Casmurro.txt
1.00
0.58
Memórias Póstumas de Brás Cubas.txt
0.68
0.35
Esaú e Jacó.txt

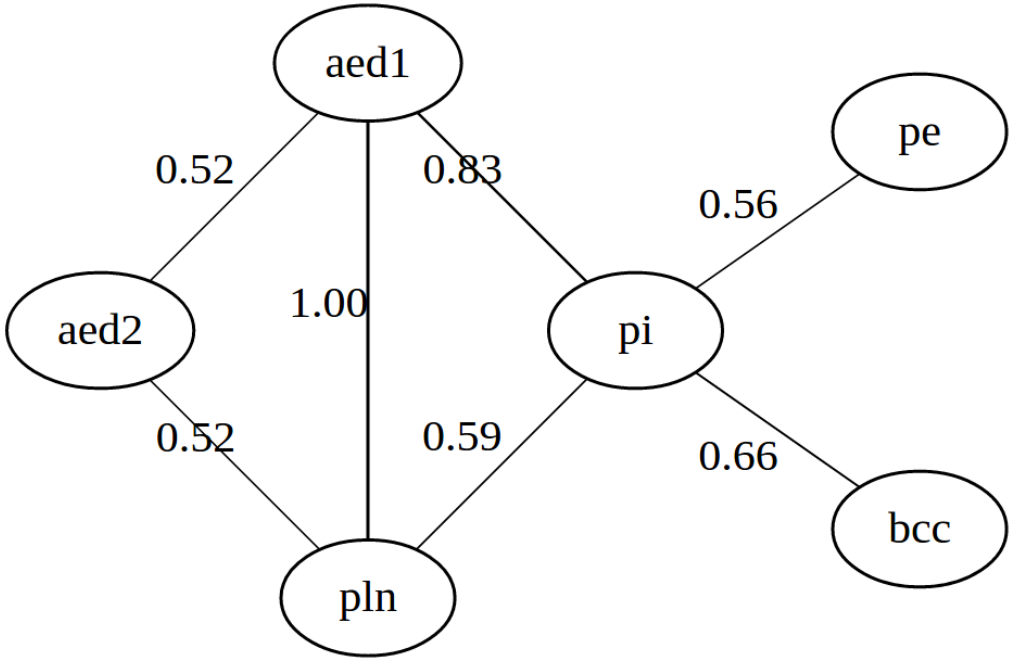

Dom Casmurro.txt
Quincas Borba.txt
1.00
0.68
0.58
Memórias Póstumas de Brás Cubas.txt
Esaú e Jacó.txt

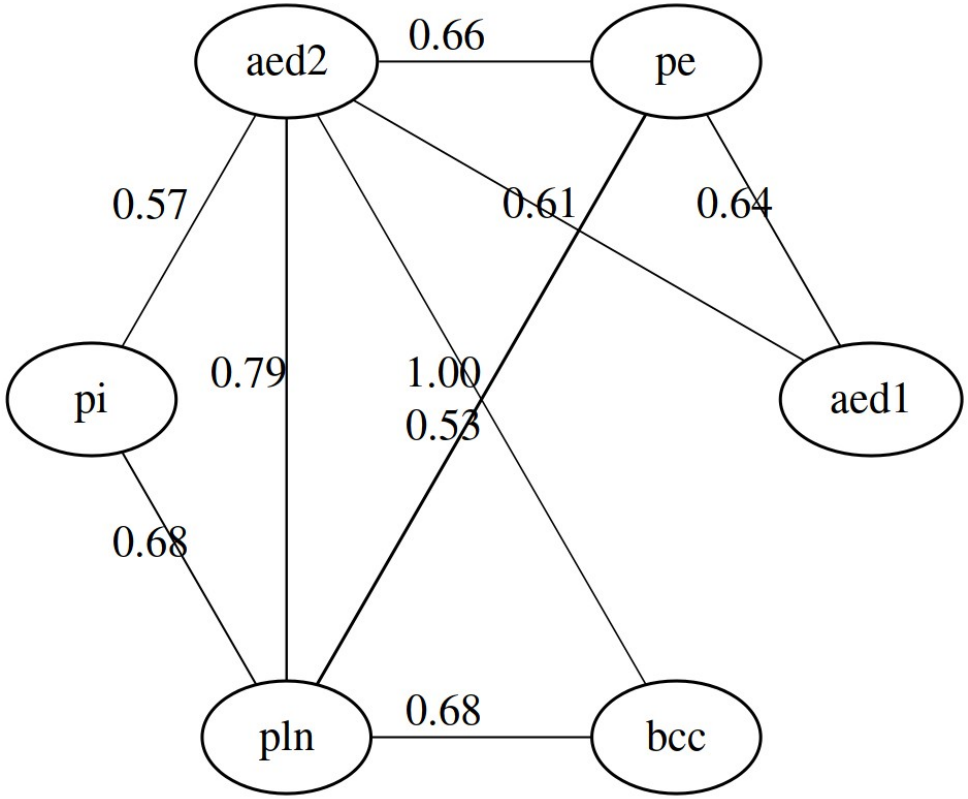
machado-db

$$\text{cosine}(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}||\vec{w}|}$$

```
# distancia Euclidiana
dist[d1,d2] = numpy.linalg.norm(M[:,d1]-M[:,d2])
# distancia (similaridade) Cosseno
dist[d1,d2] = numpy.dot(M[:,d1], M[:,d2])/(numpy.linalg.norm(M[:,d1])*numpy.linalg.norm(M[:,d2]))
```

ufabc-bcc


aed1
pe
0.52
0.83
0.56
aed2
1.00
pi
0.52
0.59
0.66
bcc
pln


aed2
0.66
pe
0.57
0.61
0.64
pi
0.79
1.00
0.53
aed1
0.68
pln
0.68
bcc

Similaridade entre palavras

Matriz de co-ocorrência: termo-documento

Duas **palavras** são similares se os vetores são similares

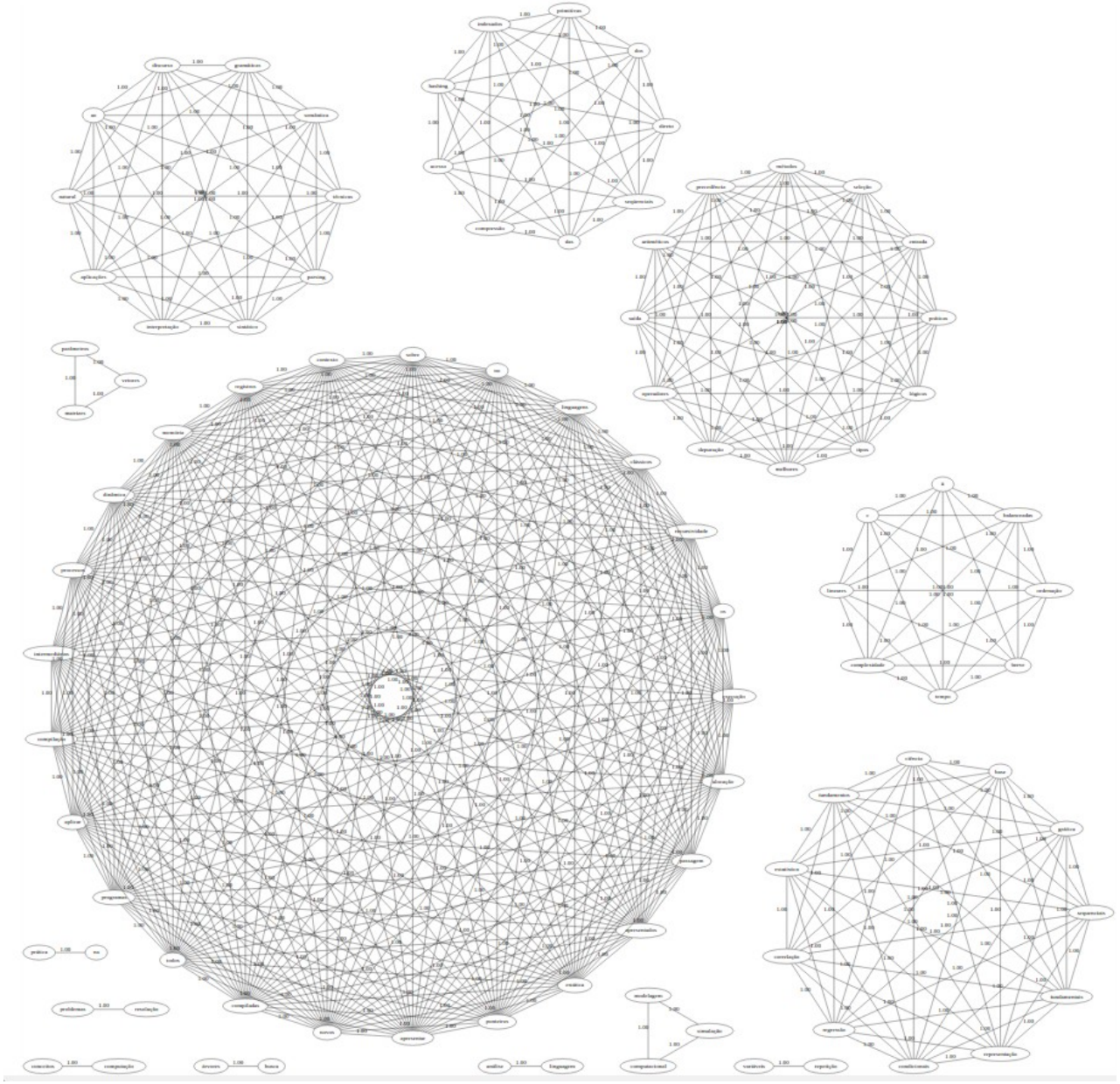
	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

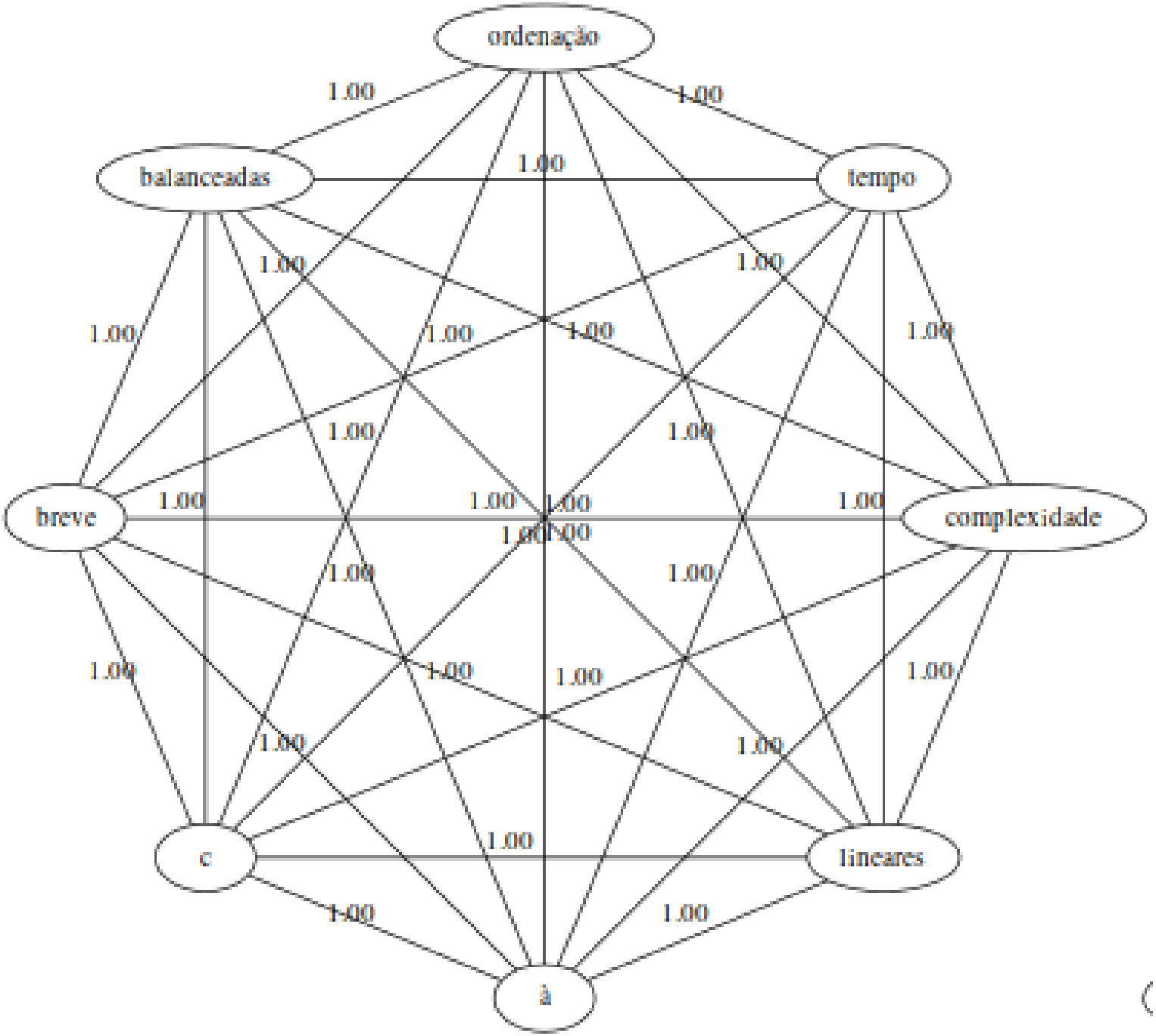
A dimensão do vetor é o número de documentos: $N^{|D|}$

teste3.py

```
# distancia entre palavras
dist = numpy.ones((V,V))*numpy.nan
for w1 in range(0, V-1):
    for w2 in range(w1+1, V):
        dist[w1,w2] = numpy.linalg.norm(M[w1,:]-M[w2,:])
print(dist)


# criando o grafo de documentos (similaridade entre documentos)
dist = 1 - (dist-numpy.nanmin(dist))/(numpy.nanmax(dist)-numpy.nanmin(dist))

txtGraph = "\ngraph{"
for w1 in range(0, V-1):
    for w2 in range(w1+1, V):
      if dist[w1,w2]!=numpy.nan and dist[w1,w2]>=0.95:
          txtGraph += '\n "{0}" -- "{1}"[label="{2:.2f}", penwidth={2:.2f}]'
txtGraph += "\n}"

print(txtGraph)
```

ufabc-bcc

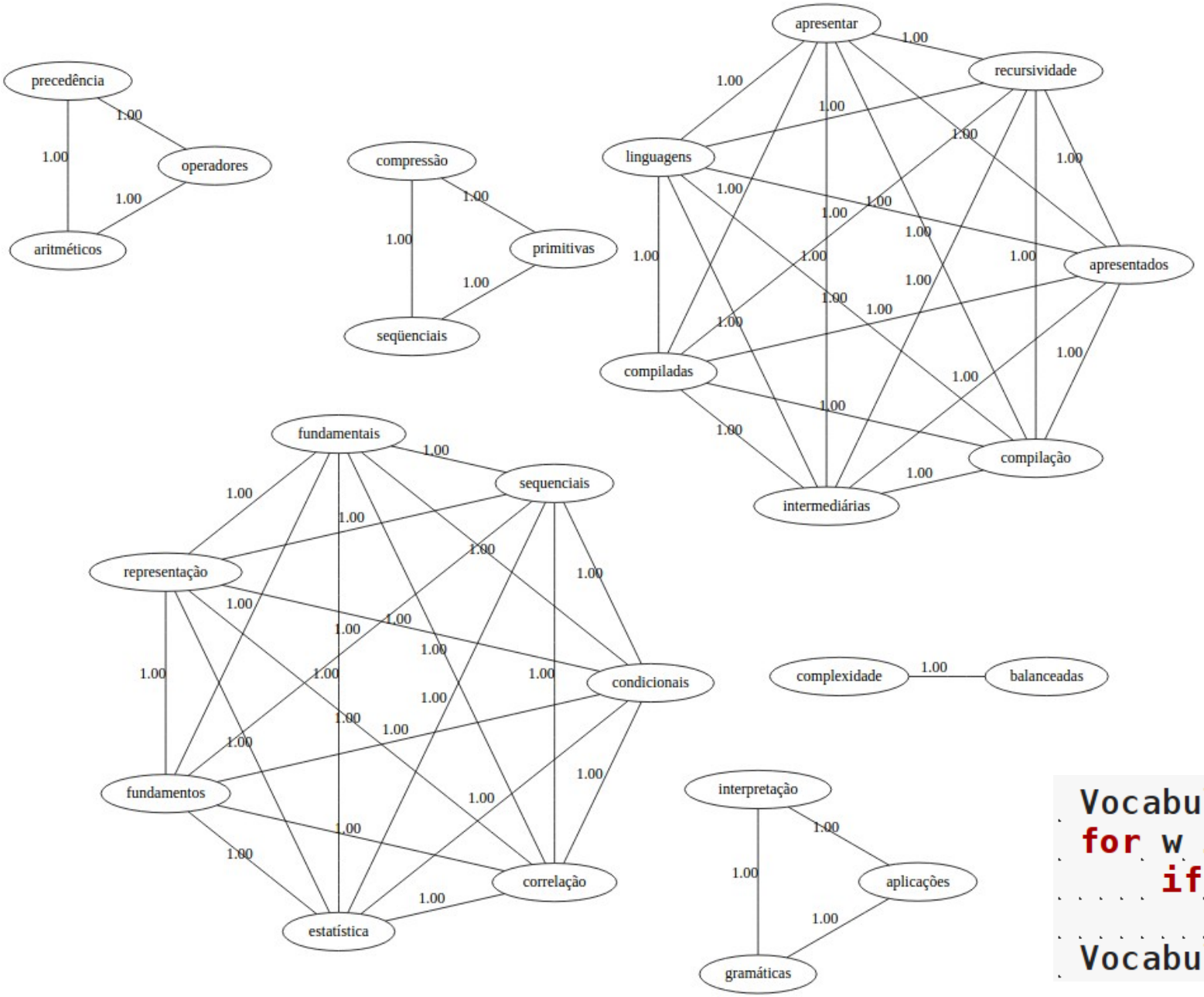



ordenação
balanceadas
tempo
breve
complexidade
c
lineares
à
1.00

Vocabulário: ufabc-bcc (109 palavras)

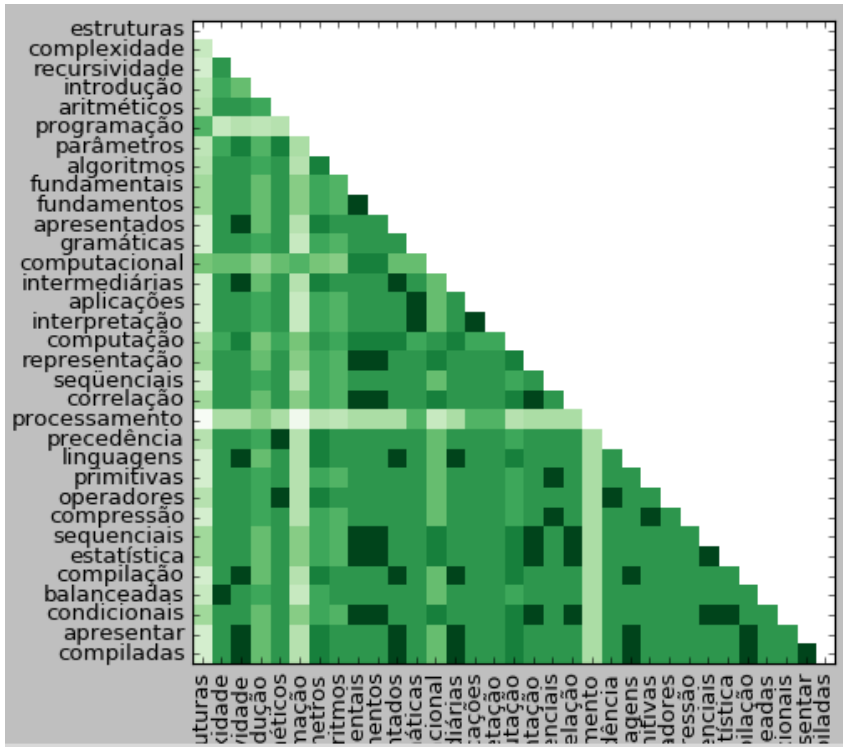
{'modelagem', 'operadores', 'indexados', 'c', 'da', 'todos', 'e',
'ao', 'seqüenciais', 'estruturas', 'apresentar', 'fundamentais',
'sobre', 'correlação', 'linguagens', 'dos', 'matrizes', 'lógicos',
'compressão', 'estatística', 'tipos', 'clássicos', 'regressão',
'busca', 'balanceadas', 'interpretação', 'parâmetros',
'computacional', 'melhores', 'técnicas', 'análise', 'alocação',
'passagem', 'seleção', 'recursividade', 'acesso', 'dados',
'conceitos', 'ponteiros', 'aritméticos', 'árvores', 'em',
'aplicações', 'programação', 'natural', 'vetores', 'repetição',
'registros', 'breve', 'apresentados', 'dinâmica', 'os', 'gráfica',
'processos', 'básicas', 'práticas', 'ciência', 'resolução',
'introdução', 'das', 'algoritmos', 'programas', 'hashing',
'condicionais', 'aplicar', 'direto', 'base', 'discurso', 'na', 'no',
'ordenação', 'lógica', 'funções', 'linguagem', 'variáveis',
'processamento', 'memória', 'novos', 'de', 'problemas', 'estática',
'sequenciais', 'fundamentos', 'lineares', 'saída', 'complexidade',
'intermediárias', 'depuração', 'computação', 'sintático', 'contexto',
'compiladas', 'simulação', 'semântica', 'prática', 'entrada',
'tempo', 'parsing', 'noções', 'arquivos', 'gramáticas', 'a',
'compilação', 'à', 'precedência', 'execução', 'primitivas',
'métodos', 'representação'}

teste4.py: ufabc-bcc

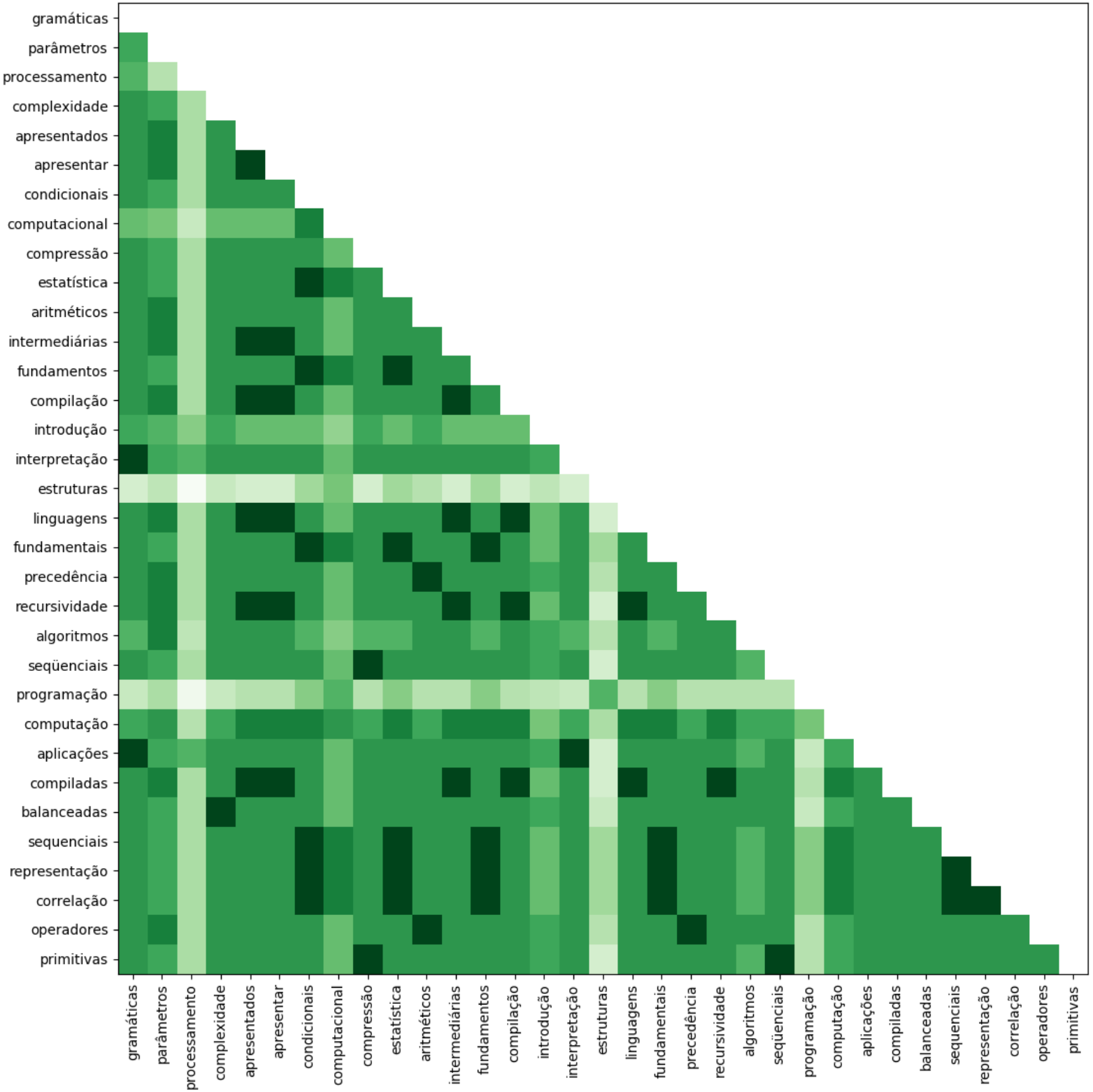



```
Vocabulary2 = set([])
for w in Vocabulary:
    if len(w)>=10:
        Vocabulary2.add(w)
Vocabulary = Vocabulary2
```

teste5.py: ufabc-bcc




```
# similaridade entre documentos
dist = 1 - (dist-numpy.nanmin(dist))/(numpy.nanmax(dist)-numpy.nanmin(dist))

plt.imshow(dist, cmap='Greens', interpolation='nearest')
plt.yticks(range(0,len(vocabulary)), vocabulary )
plt.xticks(range(0,len(vocabulary)), vocabulary )
plt.xticks(rotation=90)
plt.show()
```


gramáticas
parâmetros
processamento
complexidade
apresentados
apresentar
condicionais
computacional
compressão
estatística
aritméticos
intermediárias
fundamentos
compilação
introdução
interpretação
estruturas
linguagens
fundamentais
precedência
recursividade
algoritmos
seqüenciais
programação
computação
aplicações
compiladas
balanceadas
sequenciais
representação
correlação
operadores
primitivas

teste6.py: machado-db

```
# distancia entre palavras
dist = numpy.ones((V,V))*numpy.nan
for w1 in range(0, V):
    for w2 in range(0, V):
        if w1!=w2:
            d = numpy.linalg.norm(M[w1,:]-M[w2,:])
            dist[w1,w2] = d
            dist[w2,w1] = d
print(dist)

# similaridade entre documentos
dist = 1 - (dist-numpy.nanmin(dist))/(numpy.nanmax(dist)-numpy.nanmin(dist))

while True:
    w = input("\nDigite uma palavra: ")
    if w in vocabulary:
        i = vocabulary.index(w)
        for j in range(0,V):
            if dist[i,j]==1:
                print(vocabulary[j])
    else:
        print("palavra nao esta no vocabulario")
```

teste6.py: machado-db

```
Digite uma palavra: defendendo
separar-nos
comerciante
honestidade
respondendo
demoradamente
importância
continuando
aborrecível
desfazendo
sussurrava
chegaremos
interrompesse
ressentimentos
```

teste6.py: nlp-book

```
Digite uma palavra: datasets
thesaurus
sufficient
developed
descent
direction
itself
bigrams
digital
showed
predict
expressed
points
element
configuration
personality
combine
difficult
metrics
starting
dependent
tokenization
spanning
earliest
```

Desafio 1: Bônus +0.5 na MF

Resumo de 1 trabalho (tese/dissertação) relacionada a PLN, defendida no nível de mestrado ou doutorado em 2018.

Resumo nos mesmos moldes dos resumos de aula (detalhe a contribuição proposta pelo candidato).

Porque o trabalho vale um mestrado/doutorado?

Envio pelo tidia (seção atividades).


Catálogo de Teses e Dissertações
Busca
"processamento de linguagem natural"
Buscar
Painel de informações quantitativas (teses e dissertações)
Inicio > Busca
55 resultados para "processamento de linguagem natural"
Exibindo 1-20 de 55
<< < 1 2 3 > >>
Refinar meus resultados
Tipo: 2 opções
Mestrado (Dissertação) 46
Doutorado (Tese) 9
Ano: 1 opções
2018 55
Autor: 55 opções
ADALNIZA MOURA PUCCA DA SILVA 1
ALEX DE PAULA BARROS 1
ANDERSON PINHEIRO CAVALCANTI 1
ANGELA CRISTINA PEREIRA 1
ARNON BRUNO VENTRILHO DOS SANTOS 1
Deve selecionar um dos 55 trabalhos publicados em 2018
2. BARROS, ALEX DE PAULA. A Flexible Compositional Approach to Word Sense Disambiguation' 27/07/2018 undefined f. Mestrado em CIÊNCIAS DA COMPUTAÇÃO Instituição de Ensino: UNIVERSIDADE FEDERAL DE MINAS GERAIS, Belo Horizonte Biblioteca Depositária: Biblioteca Universitária da Universidade Federal de Minas Gerais
Detalhes
3. RODRIGUES, CHARLES. CRITÉRIOS PARA ADOÇÃO DE E-BOOK EM BIBLIOTECAS DIANTE DO PARADIGMA DA COMPUTAÇÃO NAS NUVENS' 09/03/2018 265 f. Doutorado em CIÊNCIA DA INFORMAÇÃO Instituição de Ensino: UNIVERSIDADE FEDERAL DE SANTA CATARINA, Florianópolis Biblioteca Depositária: Biblioteca Universitária
Detalhes
4. CARVALHO, LUAN ALISON CARDOSO DE. MODELAGEM DE ARQUITETURA DE SOFTWARE DE UM ROBÔ MANIPULADOR DE BLOCOS INTERATIVO POR CONVERSAÇÃO' 18/05/2018 95 f. Mestrado em ESTUDOS DE LINGUAGENS Instituição de Ensino: CENTRO FEDERAL DE EDUCAÇÃO TECN. DE MINAS GERAIS, Belo Horizonte Biblioteca Depositária: CEFET-MG CAMPUS I
Detalhes
5. BOSSOLANI, CARLOS AUGUSTO. Representações Distribuídas de Texto Aplicadas em Análise de Sentimento de Mensagens Curtas e Ruidosas' 14/12/2018 100 f. Mestrado em Ciência da Computação Instituição de Ensino: UNIVERSIDADE FEDERAL DE SÃO CARLOS, Sorocaba Biblioteca Depositária: BSo
Detalhes
6. TOHALINO, JORGE ANDONI VALVERDE. Sumarização extractiva de documentos usando redes complexas' 15/06/2018 116 f. Mestrado em CIÊNCIAS DA COMPUTAÇÃO E MATEMÁTICA COMPUTACIONAL Instituição de Ensino: UNIVERSIDADE DE SÃO PAULO (SÃO CARLOS), São Carlos Biblioteca Depositária: Prof. Achille Bassi
Detalhes
7. SANTOS, FLAVIO ARTHUR OLIVEIRA. Sobre o uso de conhecimento especialista para auxiliar no aprendizado de Word Embeddings' 31/07/2018 70 f. Mestrado em Ciência da Computação Instituição de Ensino: FUNDAÇÃO UNIVERSIDADE FEDERAL DE SERGIPE, São Cristóvão Biblioteca Depositária: Bicen UFS
Detalhes

PLATAFORMA Sucupira

Dados do Trabalhos de Conclusão

Exemplo de dissertação

Instituição de Ensino Superior:	UNIVERSIDADE DE SÃO PAULO
Programa:	Sistemas de Informação (33002010214P0)
Título:	Reconhecimento de traços de personalidade com base em textos
Autor:	BARBARA BARBOSA CLAUDINO DA SILVA
Tipo de Trabalho de Conclusão:	DISSERTAÇÃO
Data Defesa:	27/02/2018
Resumo:	Apresentamos uma pesquisa na área de Processamento de Linguagem Natural, para reconhecimento de personalidade com base em textos da língua portuguesa. Neste trabalho utilizamos textos provenientes da rede social Facebook, em conjunto com o modelo de personalidade dos Cinco Grandes Fatores, para construir um córpus rotulado com as personalidades de seus autores e, após a identificação das características mais relevantes para o reconhecimento de personalidade, construir modelos computacionais utilizando essas características. Utilizando-se métodos provenientes de léxicos, como o dicionário LIWC ou atributos psicolinguísticos, e métodos provenientes do próprio texto, como bag of words, representação distribuída de palavras e de documentos foram desenvolvidos modelos para reconhecimento de personalidade sem a necessidade de outros métodos mais comumente utilizados para essa tarefa, como inventários ou entrevistas com psicólogos. Os resultados dos métodos de representação distribuída são ligeiramente superiores do que os resultados utilizando o dicionário LIWC, com a vantagem de não exigirem recursos dependentes de um idioma específico.
Palavras-Chave:	Big Five;Modelo dos cinco grandes fatores;Personalidade;Processamento de linguagem natural
Abstract:	We present a research proposal in the Natural Language Processing field, to recognize personality through texts in the portuguese language. Using texts from the social network Facebook we built a corpus labeled with authors Big-5 personality traits, and after identifying the most relevant atributes to recognize personality, we built computational models based on those attributes. The model was expected to recognize personality without the help of any other methods commonly used in this task, such as inventories or interviews with psychologists. Using lexical methods such as the LIWC dictionary or psycholinguistic attributes, and methods from the text itself, such as bag of words, distributed representation of words and documents, we obtained models for personality recognition without the need of other methods most commonly used for this task. The results of distributed representation methods are slightly better than the results using the LIWC dictionary, with the advantage of not requiring features dependent on a specific language.
Keyword:	Big Five;Big five model;Natural language processing;Personality
Volume:	1
Páginas:	98
Idioma:	PORTUGUES
Biblioteca Depositária:	UNIVERSIDADE DE SÃO PAULO
Anexo:	corrigida.pdf

O documento disponível para leitura.

Contexto

Área de Concentração:	METODOLOGIA E TÉCNICAS DA COMPUTAÇÃO
Linha de Pesquisa:	INTELIGÊNCIA DE SISTEMAS
Projeto de Pesquisa:	-

Banca Examinadora

Orientador:	IVANDRE PARABONI
O orientador principal compôs a banca do discente?	Sim

Nome	Categoria
ARIADNE MARIA BRITO RIZZONI CARVALHO	Participante Externo
ARIANE MACHADO LIMA	Docente
FERNANDO FAGUNDES FERREIRA	Participante Externo