



Processamento de Linguagem Natural

Processamento Básico de Texto

Expressões Regulares e Normalização de Texto

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor César Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Processamento de Texto Básico

Expressões Regulares



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Expressões Regulares

- Uma linguagem formal para especificar cadeias de caracteres
- Como podemos encontrar qualquer das opções abaixo?
 - woodchuck*
 - woodchucks
 - Woodchuck
 - Woodchucks



*Marmota (Um tipo grande de esquilo)



Expressões Regulares: Disjunções

- Letras dentro dos colchetes []

Padrão	Equivale
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	Any digit

- Conjuntos em intervalos [A-Z]

Padrão	Equivale	
[A-Z]	Uma letra maiúscula	<u>D</u> renched Blossoms
[a-z]	Uma letra minúscula	<u>m</u> y beans were impatient
[0-9]	Um único dígito	Chapter <u>1</u> : Down the Rabbit Hole



Expressões Regulares: Negação em Disjunções

- Negações **[^Ss]**
 - ^ significa negação apenas quando vem primeiro nos []

Padrão	Equivale	
[^A-Z]	Uma letra “not” maiúscula	O <u>y</u> fn pripetchik
[^Ss]	Não ‘S’ e não ‘s’	<u>I</u> have no exquisite reason
[^e^]	Não ‘e’ e não ‘^’	Look h <u>e</u> re
a^b	O padrão a ^ b	Look up <u>a^b</u> now



Expressões Regulares: Mais Disjunções

- Woodchuck e Groundhog são sinônimos para Marmota!
- Barra vertical | para disjunção

Padrão	Equivale
groundhog woodchuck	
yours mine	yours mine
a b c	= [abc]
[gG]roundhog [Ww]oodchuck	



Photo D. Fletcher



Expressões Regulares: ? * + .

Padrão	Equivale	
colou?r	Caractere anterior opcional	<u>color</u> <u>colour</u>
oo*h!	0 ou mais do caractere anterior	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
o+h!	1 ou mais do caractere anterior	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
baa+	``	<u>baa</u> <u>baaa</u> <u>baaaa</u> <u>baaaaa</u>
beg.n	Qualquer* caractere	<u>begin</u> <u>begun</u> <u>begun</u> <u>beg3n</u>

*exceto quebra de linha

7



Stephen C Kleene

Kleene *, Kleene +



Expressões Regulares: Ancoras: ^ \$

Padrão	Equivale	
^ [A-Z]	Inicia com letra maiúscula	<u>P</u> alo Alto
^ [^A-Za-z]	Inicia com algo que não seja letra maiúscula ou minúscula	<u>1</u> <u>"Hello"</u>
\. \$	Finaliza com ponto	The end <u>.</u>
. \$	Finaliza com algum caractere	The end <u>?</u> The end <u>!</u>



Exemplo

- Encontrar todas as instâncias da palavra “the” em um texto.

the

Perde exemplos com letras maiúsculas

[tT]he

Retorna incorretamente other ou theology

[^a-zA-Z][tT]he[^a-zA-Z]



Tipos de Erros

- O processo que acabamos de vicenciar foi baseado na resolução de dois tipos de erros
 - Encontrar instâncias que não deveriam ser encontradas (there, then, other)
 - Falsos positivos (Tipo I)
 - Não encontrar instâncias que deveriam ser encontradas (The)
 - Falso negativo (Tipo II)



Tipos de Erros

- Em PLN nós estamos sempre lidando com esses tipos de erros.
- Reduzir a taxa de erro para uma aplicação sempre envolve dois esforços antagônicos:
 - Melhorar acurácia ou precisão (minimizando falsos positivos)
 - Melhorar cobertura (minimizando falsos negativos).



Sumário

- Expressões Regulares desempenham uma função surpreendentemente ampla
 - Sequencias sofisticadas de expressões regulares são sempre o primeiro modelo para qualquer processamento de texto
- Para muitas tarefas difíceis, nós usamos classificadores da aprendizagem de maquina
 - Expressões regulares são usadas como características nos classificadores
 - Pode ser bem útil na identificação de generalizações



Processamento de Texto Básico

Expressões Regulares



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Processamento de Texto Básico

Separação de Tokens



Normalização de Texto

- Todas atividades de PLN precisam fazer normalização de texto:
 1. Segmentação/tokenizing de palavras em texto em corrido
 2. Normalização do formato das palavras
 3. Segmentação de sentenças no texto em execução



Quantas palavras?

- I do uh main- mainly business data processing
 - Fragmentos, pausas preenchidas
- Seuss's **cat** in the hat is different from other **cats**!
 - **Lema**: palavras com mesmo tronco (forma canônica)
 - **cat** and **cats** = mesmo lema
 - **Forma**: Flexões de um lema
 - **cat** and **cats** = mesmo lema, formas diferentes
 - **Morfema**: Elemento da linguagem que carrega significado
 - **Desumidificar**: três morphemas “**des**”, “**umidi**”, “**ficar**”



Quantas palavras?

they lay back on the San Francisco grass and looked at the stars and their

- **Tipo:** um elemento do vocabulário.
- **Token:** uma instância daquele tipo no texto em execução.
- Quantos?
 - 15 tokens (ou 14)
 - 13 tipos (ou 12) (ou 11?)



Quantas palavras?

N = número de tokens

Church and Gale (1990): $|V| < O(N^{\frac{1}{2}})$

V = vocabulário = conjunto de tipos

$|V|$ é o tamanho do vocabulário

	Tokens = N	Type= $ V $
Switchboard phone conversations	2.4 milhões	20 mil
Shakespeare	884,000	31 mil
Google N-grams	1 trilhões	13 mil



Separação de Tokens Simples no UNIX

- (Inspirado em “UNIX para Poetas” de Ken Church.)
- Dado um arquivo de texto, devolva os tokens e suas frequências

```
tr -sc 'A-Za-z' '\n' < shakes.txt
```

Mudar cada não-alfa para nova linha

```
| sort
```

Ordenar em ordem alfabética

```
| uniq -c
```

Juntar e contar cada tipo

```
1945 A          25 Aaron
 72 AARON       6 Abate
19 ABBESS       1 Abates
 5 ABBOT        5 Abbess
               6 Abbey
... ..         3 Abbot
19
```

```
.... ..
```



Primeiro Passo: Separação em Tokens

```
tr -sc 'A-Za-z' '\n' < shakes.txt | head
```

```
THE  
SONNETS  
by  
William  
Shakespeare  
From  
fairest  
creatures  
We  
...
```



Segundo Passo: Ordenação

```
tr -sc 'A-Za-z' '\n' < shakes.txt | sort | head
```

A
A
A
A
A
A
A
A
A
...



Por fim: Contagem

- Unir maiúsculas e minúsculas

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c
```

- Ordenar as contagens

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c | sort -n -r
```

```
23243 the
22225 i
18618 and
16339 to
15687 of
12780 a
12163 you
10839 my
10005 in
8954 d
```

O que aconteceu aqui?



Questões na Separação de Tokens

- Finland's capital → Finland, Finlands, Finland's ?
- what're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two?
- m.p.h., PhD. → ??



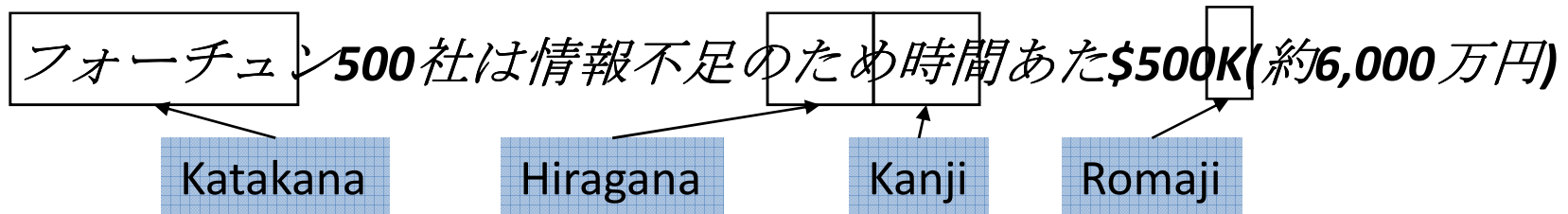
Separação de Tokens: Questões de Idioma

- Francês
 - *L'ensemble* → um token ou dois?
 - *L ? L' ? Le ?*
 - Queremos que *l'ensemble* seja equivalente a *un ensemble*
- Substantivos compostos em Alemão não são segmentados
 - *Lebensversicherungsgesellschaftsangestellter*
 - ‘Funcionário de uma companhia de seguros de vida’
 - Recuperação de informação em Alemão precisa de um **separador de compostos**



Separação de Tokens: Questões de Idioma

- Chinês e Japonês não têm espaços entre palavras:
 - 莎拉波娃现在居住在美国东南部的佛罗里达。
 - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
 - Sharapova now lives in US southeastern Florida
- Ainda mais complicado em Japonês, por ter múltiplos alfabetos misturados
 - Datas/Quantidades em múltiplos formatos



25 Usuário final pode expressar uma pergunta inteiramente em hiragana!



Separação de Tokens em Chinês

- Também chamado de **Separação em Segmentos**
- Palavras em Chinês são compostas de caracteres
 - Caracteres geralmente representam 1 sílaba e 1 morfema.
 - Uma palavra tem em média 2.4 caracteres.
- Algoritmo inicial padrão de segmentação:
 - Maximum Matching (também conhecido como Greedy)



Maximum Matching

Algoritmo de Separação em Segmentos

- Dada uma lista de palavras de Chinês e uma cadeia de caracteres.
 - 1) Comece com um apontador no começo da cadeia de caracteres
 - 2) Encontre a palavra mais longa no dicionário que corresponde a sequência de caracteres a partir do apontador
 - 3) Mova o ponteiro através da palavra
 - 4) Volte ao passo 2



Segmentação Max-match: Ilustração

- Thecatinthehat the cat in the hat
- Thetabledownthere the table down there
 theta bled own there
- Geralmente não funciona em Inglês!
- Mas funciona surpreendentemente bem em Chinês
 - 莎拉波娃现在居住在美国东南部的佛罗里达。
 - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
- Algoritmos modernos de segmentação probabilística são ainda melhores



Processamento de Texto Básico

Separação de Tokens



Processamento de Texto Básico

Normalização de
Palavras e Stemming



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Normalização

- Necessidade por normalizar termos
 - Recuperação de Informação: texto indexado & termos de busca precisam ter a mesma forma.
 - Queremos combinar **U.S.A.** e **USA**
- Nós implicitamente definimos classes equivalentes de termos
 - e.g., deletando pontos em um termo
- Alternativa: expansão assimétrica:
 - Enter: **window** Search: **window, windows**
 - Enter: **windows** Search: **Windows, windows, window**
 - Enter: **Windows** Search: **Windows**
- Potencialmente mais poderoso, mas menos eficiente



Case Folding

- Aplicações como IR: transformam todas as letras em minúsculas
 - Já que usuários tendem a usar minúsculas
 - Possível Exceção: maiúsculas no meio da sentença?
 - e.g., *General Motors, Fed* vs. *fed*
 - *SAIL* vs. *sail*
- Para análise de sentimentos, MT, Extração de informação
 - Ter letras maiúsculas e minúsculas é útil(*US* versus *us* é importante)



Lemmatização

- Reduzir inflexões ou formas variantes para a forma base
 - *am, are, is* → *be*
 - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatização: tem que encontrar forma base correta no dicionário
- Tradução automática
 - Espanhol *quiero* ('I want'), *quieres* ('you want') mesmo lemma que *querer* 'want'



Morfologia

- **Morfemas:**
 - A menores unidades de significado que fazem palavras
 - **Stems:** Unidades de significado base
 - **Afixos:** Pedacos que aderem à stems
 - Sempre com funções gramaticais



Stemming

- Reduz termos à seus “caules” em recuperação de informação
- *Stemming* é o corte de afixos
 - Dependente de linguagem
 - e.g., ***automate(s), automatic, automation*** todos reduzidos para ***automat.***

*for example compressed
and compression are both
accepted as equivalent to
compress.*



for exampl compress and
compress ar both accept
as equival to compress



Algoritmo de Porter

O mais comum Stemmer de Inglês

Step 1a

sses → ss	caresses → caress
ies → i	ponies → poni
ss → ss	caress → caress
s → ∅	cats → cat

Step 1b

(*v*)ing → ∅	walking → walk
	sing → sing
(*v*)ed → ∅	plastered → plaster
...	

Step 2 (para stems longos)

ational → ate	relational → relate
lizer → ize	digitizer → digitize
Ator → ate	operator → operate
...	

Step 3 (para stems mais longos)

al → ∅	revival → reviv
able → ∅	adjustable → adjust
ate → ∅	activate → activ
...	



Visualizando Morfologia como um corpo

Por quê só retirar –ing se tiver uma vogal?

$(*v^*)ing \rightarrow \emptyset$ walking \rightarrow walk
 sing \rightarrow sing



Visualizando Morfologia como um corpo

Por quê só retirar –ing se tiver uma vogal?

$(*v^*)ing \rightarrow \emptyset$ walking \rightarrow walk
sing \rightarrow sing

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep 'ing$' | sort | uniq -c | sort -nr
```

1312 King	548 being
548 being	541 nothing
541 nothing	152 something
388 king	145 coming
375 bring	130 morning
358 thing	122 having
307 ring	120 living
152 something	117 loving
145 coming	116 Being
130 morning	102 going

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep '[aeiou].*ing$' | sort | uniq -c | sort -nr
```



Lidar com morfologia complexa as vezes é necessário

- Alguns idiomas necessitam de segmentação complexa de morfemas
 - Turco
 - **Uygarlastiramadiklarimizdanmissinizcasina**
 - ‘(comportar-se) como se você estivesse entre aqueles que nós não pudemos civilizar’
 - **Uygar** ‘civilized’ + **las** ‘become’
 - + **tir** ‘cause’ + **ama** ‘not able’
 - + **dik** ‘past’ + **lar** ‘plural’
 - + **imiz** ‘p1pl’ + **dan** ‘abl’
 - + **mis** ‘past’ + **siniz** ‘2pl’ + **casina** ‘as if’



Processamento de Texto Básico

Normalização de
Palavras e Stemming



Processamento de Texto Básico

Segmentação de
Sentenças e Árvores de
Decisão

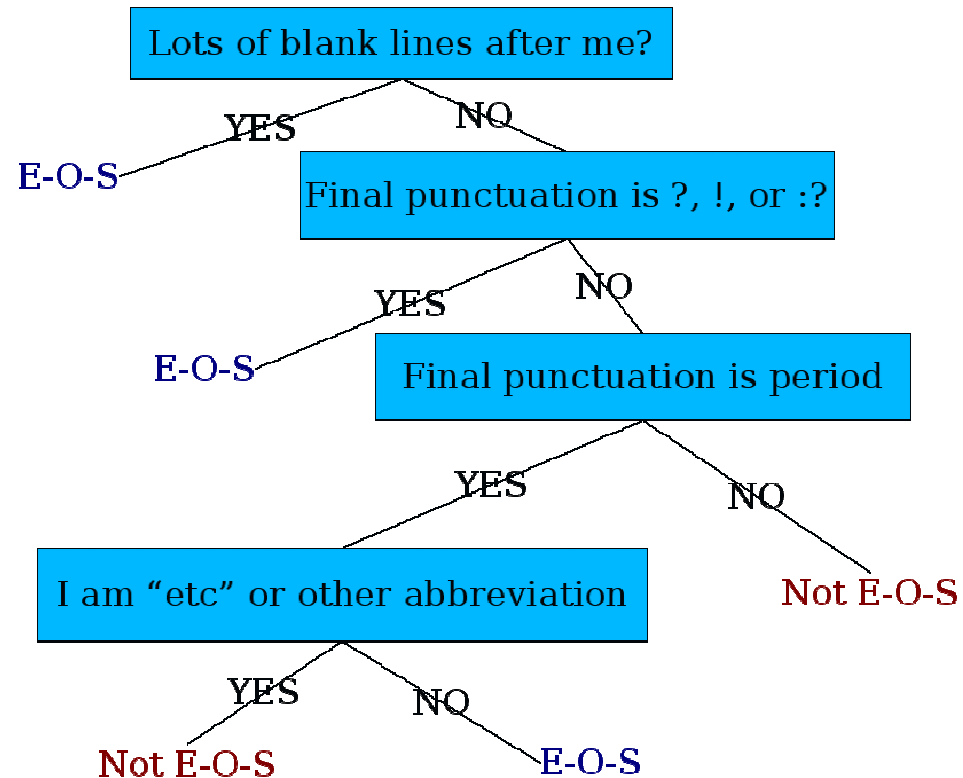


Segmentação de Sentenças

- !, ? São relativamente não ambíguos
- “.” É bastante ambíguo
 - Delimitador de sentença
 - Abreviações como Inc. ou Dr.
 - Números como .02% ou 4.3
- Construir um classificador binário
 - Observa o “.”
 - Decide entre FimDeSentença/NãoFimDeSentença
 - Classificadores: baseado em regras escritas à mão, expressões regulares, ou aprendizagem de máquina



Determinar se uma palavra é fim de sentença: a Decision Tree





Características de Árvores de Decisão mais Sofisticadas

- Caso de palavra com “.”: Upper, Lower, Cap, Number
- Caso de palavra depois de “.”: Upper, Lower, Cap, Number
- Características Numéricas
 - Tamanho da palavra terminando com “.”
 - Probabilidade (palavra com “.” ocorrer no fim da sentença)
 - Probabilidade (palavra depois de “.” ocorrer no começo da sentença)



Implementando Árvores de Decisão

- Uma árvore de decisão é baseada em declarações *if-then-else*
- A parte interessante é escolher as características
- Configurar a estrutura é sempre muito difícil de fazer manualmente
 - Configuração manual é apenas viável para características de domínios muito simples
 - Para características numéricas é muito difícil escolher cada limiar
 - Ao invés disso, a estrutura é normalmente aprendida por aprendizagem de máquina com uma base de treinamento



Ávores de Decisão e Outros classificadores

- Nós podemos configurar as características em uma árvore de decisão
- As características podem ser exploradas por qualquer tipo de classificador
 - Regressão logística
 - SVM
 - Redes Neurais
 - etc.



Processamento de Texto Básico

Segmentação de
Sentenças e Árvores de
Decisão



Processamento de Linguagem Natural

Distância Mínima de Edição

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor César Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Distância Mínima de Edição

Definição de Distância Mínima de Edição



O quão similar duas coisas são?

- Correção Ortográfica

- O usuário digitou “graffe”

Qual é o mais próximo?

- graf
 - graft
 - grail
 - giraffe

- Computação Biológica

- Alinhar duas sequências de nucleotídeos

AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTCGATTTGCCCGAC

- Alinhamento resultante:

– AGGCTATCACCTGACCTCCAGGCCGA – – TGCCC – – –
TAG – CTATCAC – – GACCGC – – GGTCGATTTGCCCGAC

- Também para Tradução Automática, Extração de Informação, Reconhecimento de Discurso



Distância de Edição

- Distância mínima de edição entre duas cadeias de caracteres
- É o número mínimo de operações de edição
 - Inserção (i)
 - Deleção (d)
 - Substituição (s)
- Necessário para transformar um no outro



Distância Mínima de Edição

- Duas cadeias de caracteres e seu alinhamento:

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N



Distância Mínima de Edição

I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s i s

- Se cada operação tem o custo de 1
 - A distância entre eles é 5
- Se substituições custam 2 (Levenshtein)
 - A distância entre eles é 8



Alinhamento em Computação Biológica

- Dadas duas sequencias de bases:

AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTTCGATTGCCCCGAC

- Alinhar cada letra a uma letra ou lacuna:

–AGGCTATCACCTGACCTCCAGGCCGA–TGCCC–
TAG–CTATCAC–GACCGC–GGTCGATTGCCCCGAC



Outros Usos de Distância de Edição em PLN

- Avaliar Tradução Automática e Reconhecimento de Discurso

R Spokesman confirms senior government adviser was shot

H Spokesman said the senior adviser was shot dead

S I D I

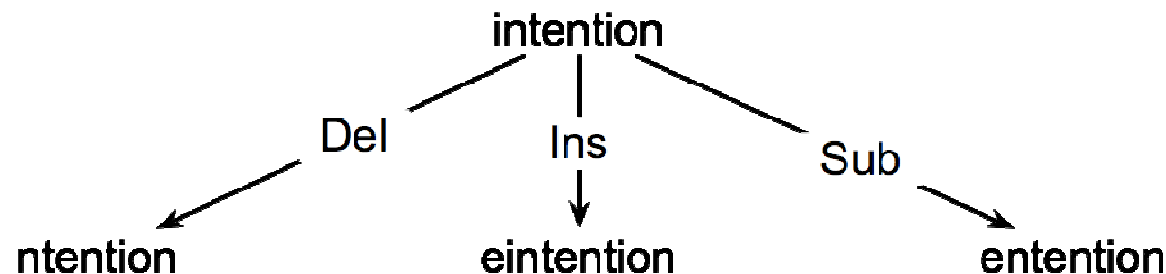
- Extração de Entidade Nomeada e Correferência de Entidade

- IBM Inc. announced today
- IBM profits
- Stanford President John Hennessy announced yesterday
- for Stanford University President John Hennessy



Como Encontrar Distância Mínima de Edição?

- Procurar por um caminho (sequência de edição) da cadeia de caracteres inicial até a final:
 - **Estado Inicial:** a sequência que está sendo transformada
 - **Operadores:** inserção deleção ou substituição
 - **Estado Objetivo:** a sequência que estamos tentando obter
 - **Custo do caminho:** o que nós queremos minimizar: o número de edições





Edição Mínima como Busca

- Mas o espaço de todas as sequências de edição é enorme!
 - Nós não podemos nos dar o luxo de navegar ingenuamente
 - Muitos caminhos distintos acabam no mesmo estado.
 - Nós não precisamos manter todos os caminhos
 - Apenas o caminho mais curto para cada um dos estados visitados.



Definição de Distância Mínima de Edição

- Para duas cadeias de caractere
 - X de tamanho n
 - Y de tamanho m
- Nós definimos $D(i,j)$
 - A distância de edição entre $X[1..i]$ e $Y[1..j]$
 - i.e., os primeiros i caracteres de X e os primeiros j caracteres de Y
 - A distância de edição entre X e Y é, portanto: $D(n,m)$



Distância Mínima de Edição

Definição de Distância Mínima de Edição



Distância Mínima de Edição

Computando Distância
Mínima de Edição



Programação Dinâmica para Distância Mínima de Edição

- **Programação Dinâmica:** Um cálculo tabular de $D(n,m)$
- Resolver problemas combinando soluções para subproblemas.
- Bottom-up
 - Calculamos $D(i,j)$ para i,j pequenos
 - E calculamos $D(i,j)$ maiores baseados nos valores menores computados anteriormente
 - i.e., calcular $D(i,j)$ para todo i ($0 < i < n$) e j ($0 < j < m$)



De onde veio o nome Programação Dinâmica?

...Os anos 50 não foram bons anos para pesquisa matemática. O Secretário de Defesa ...tinha um medo patológico e odiava a palavra pesquisa...

Eu portanto decidi utilizar a palavra, “**Programação**”.

Eu queria passar a ideia de que era dinâmico, multiestágio... eu pensei, vamos ... escolher uma palavra que tem um significado absolutamente preciso, ou seja, **dinâmico**... é impossível usar a palavra, **dinâmico**, em um sentido pejorativo. Tente pensar em alguma combinação que vai dar um sentido pejorativo. É impossível.

Assim, eu pensei que programação dinâmica era um bom nome. Era algo que nem mesmo um congressista poderia opor-se.”

Richard Bellman, “Eye of the Hurricane: an autobiography” 1984.



Definindo Distância Mínima de Edição (Levenshtein)

- Inicialização

$$D(i, 0) = i$$

$$D(0, j) = j$$

- Relação de Recorrência:

For each $i = 1..M$

For each $j = 1..N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

- Término:

$D(N, M)$ é a distância mínima de edição

Tabela de Distância de Edição

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Tabela de Distância de Edição

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
i/j	#	E	X	E	C	U	T	I	O	N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$




Tabela de Distância de Edição

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
i/j	#	E	X	E	C	U	T	I	O	N



Distância Mínima de Edição

Computando Distância
Mínima de Edição



Distância Mínima de Edição

Backtrace para Cálculo de Alinhamentos



Calculo de Alinhamentos

- Distância de Edição não é suficiente
 - Nós frequentemente precisamos **alinhar** cada caractere das duas cadeias, um com o outro
- Nós fazemos isso mantendo um “backtrace”
- Toda vez que entramos em uma célula, lembramos de onde viemos
- Quando chegarmos ao fim,
 - Seguimos o caminho de volta a partir do canto superior direito para ler o **alinhamento**

Distância de Edição

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

MinEdit com Backtrace

n	9	↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙←↓ 12	↓ 11	↓ 10	↓ 9	↙ 8	
o	8	↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↓ 10	↓ 9	↙ 8	← 9	
i	7	↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	↙ 8	← 9	← 10	
t	6	↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙ 8	← 9	← 10	←↓ 11	
n	5	↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙↓ 10	
e	4	↙ 3	← 4	↙← 5	← 6	← 7	←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	
t	3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙ 7	←↓ 8	↙←↓ 9	↓ 8	
n	2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↓ 7	↙←↓ 8	↙ 7	
i	1	↙←↓ 2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙ 6	← 7	← 8	
#	0	1	2	3	4	5	6	7	8	9	
	#	e	x	e	c	u	t	i	o	n	



Adicionando Backtrace à Distância Mínima de Edição

- Condições iniciais:

$$D(i, 0) = i$$

$$D(0, j) = j$$

- Término:

$$D(N, M) \text{ is distance}$$

- Relação de Recorrência:

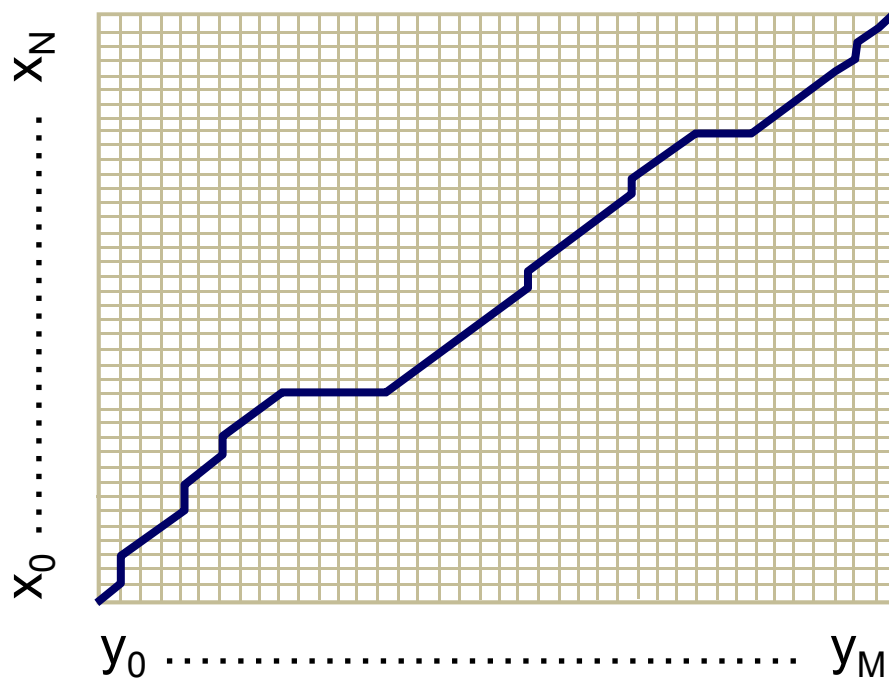
For each $i = 1 \dots M$

For each $j = 1 \dots N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 & \text{deleção} \\ D(i, j-1) + 1 & \text{inserção} \\ D(i-1, j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) & \text{substituição} \\ 0; & \text{if } X(i) = Y(j) & \text{casamento} \end{cases} \end{cases}$$
$$\text{ptr}(i, j) = \begin{cases} \text{LEFT} & \text{insertion} \\ \text{DOWN} & \text{deletion} \\ \text{DIAG} & \text{substitution} \end{cases}$$



Matriz de Distância



Todo caminho não-decrescente

de $(0,0)$ para (M, N)

corresponde a um alinhamento
de duas sequencias

Um alinhamento ótimo é composto
de subalinhamentos ótimos



Resultado do Backtrace

- Duas cadeias de caracteres e seu **alinhamento**:

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N



Performance

- Time:
 $O(nm)$
- Space:
 $O(nm)$
- Backtrace
 $O(n+m)$



Distância Mínima de Edição

Backtrace para Cálculo de Alinhamentos



Distância Mínima de Edição

Distância Mínima de Edição com Pesos



Distância de Edição com Pesos

- Por que adicionaríamos pesos ao cálculo?
 - Correção Ortográfica: algumas letras são mais prováveis de serem digitadas incorretamente do que outras
 - Biologia: certos tipos de remoção e inserção são mais prováveis que outros

Matriz de Confusão para Erros de Ortografia

sub[X, Y] = Substitution of X (incorrect) for Y (correct)

X	Y (correct)																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0
b	0	0	9	9	2	2	3	1	0	0	0	5	11	5	0	10	0	0	2	1	0	0	8	0	0	0
c	6	5	0	16	0	9	5	0	0	0	1	0	7	9	1	10	2	5	39	40	1	3	7	1	1	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	3	0	1	0	43	30	22	0	0	4	0	2	0
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0
f	0	15	0	3	1	0	5	2	0	0	0	3	4	1	0	0	0	6	4	12	0	0	2	0	0	0
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0
i	103	0	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0
j	0	1	1	9	0	0	1	0	0	0	0	2	1	0	0	0	0	5	0	0	0	0	0	0	0	0
k	1	2	8	4	1	1	2	5	0	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	3
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	11	10	2	0	0	0	0	0	0
m	1	3	7	8	0	2	0	6	0	0	4	4	0	180	0	6	0	0	9	15	13	3	2	2	3	0
n	2	7	6	5	3	0	1	19	1	0	4	35	78	0	0	7	0	28	5	7	0	0	1	2	0	2
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	4	14	39	0	0	0	18	0
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0
q	0	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	0	15	0	0	5	3	20	1
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6
u	20	0	0	0	44	0	0	0	64	0	0	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0
v	0	0	7	0	0	3	0	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0
w	2	2	1	0	1	0	0	2	0	0	1	0	0	0	0	7	0	6	3	3	1	0	0	0	0	0
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0
y	0	0	2	0	15	0	1	7	15	0	0	0	2	0	6	1	0	7	36	8	5	0	0	1	0	0
z	0	0	0	7	0	0	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0





Distância Mínima de Edição com Pesos

- Inicialização:

$$D(0,0) = 0$$

$$D(i,0) = D(i-1,0) + \text{del}[x(i)]; \quad 1 < i \leq N$$

$$D(0,j) = D(0,j-1) + \text{ins}[y(j)]; \quad 1 < j \leq M$$

- Recorrência:

$$D(i,j) = \min \begin{matrix} D(i-1,j) & + & \text{del}[x(i)] \\ D(i,j-1) & + & \text{ins}[y(j)] \\ D(i-1,j-1) & + & \text{sub}[x(i),y(j)] \end{matrix}$$

- Término:

$D(N,M)$ é a distância



Distância Mínima de Edição

Distância Mínima de Edição com Pesos



Distância Mínima de Edição

Distância Mínima de
Edição em Computação
Biológica



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

Alinhamento de Sequências

AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTCGATTGCCCCGAC

–AGGCTATCACCTGACCTCCAGGCCGA–TGCCC– –
TAG–CTATCAC–GACCGC–GGTCGATTGCCCCGAC



Por que alinhamento de sequências?

- Comparar genes ou regiões de diferentes espécies
 - Para encontrar regiões importantes
 - Determinar função
 - Encontrar forças evolucionárias
- Montar fragmentos para sequenciar DNA
- Comparar indivíduos à procura de mutações



Alinhamento em Dois Campos

- Em Processamento de Linguagem Natural
 - Nós geralmente falamos de distância (minimizada)
 - E pesos
- Em Computação Biológica
 - Nós geralmente falamos de similaridade (maximizada)
 - E scores



The Needleman-Wunsch Algorithm

A matriz contém scores (match, mismatch, gap)

Needleman-Wunsch

match = 1 mismatch = -1 gap = -1

		G	C	A	T	G	C	U
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	0	-1	-2	-3	-4	-5
A	-2	0	0	1	0	-1	-2	-3
T	-3	-1	-1	0	2	1	0	-1
T	-4	-2	-2	-1	1	1	0	-1
A	-5	-3	-3	-1	0	0	0	-1
C	-6	-4	-2	-2	-1	-1	1	0
A	-7	-5	-3	-1	-2	-2	0	0

- Inicialização:

$$D(i, 0) = -i * d \quad (d: \text{penalização})$$

$$D(0, j) = -j * d$$

- Relação de Recorrência:

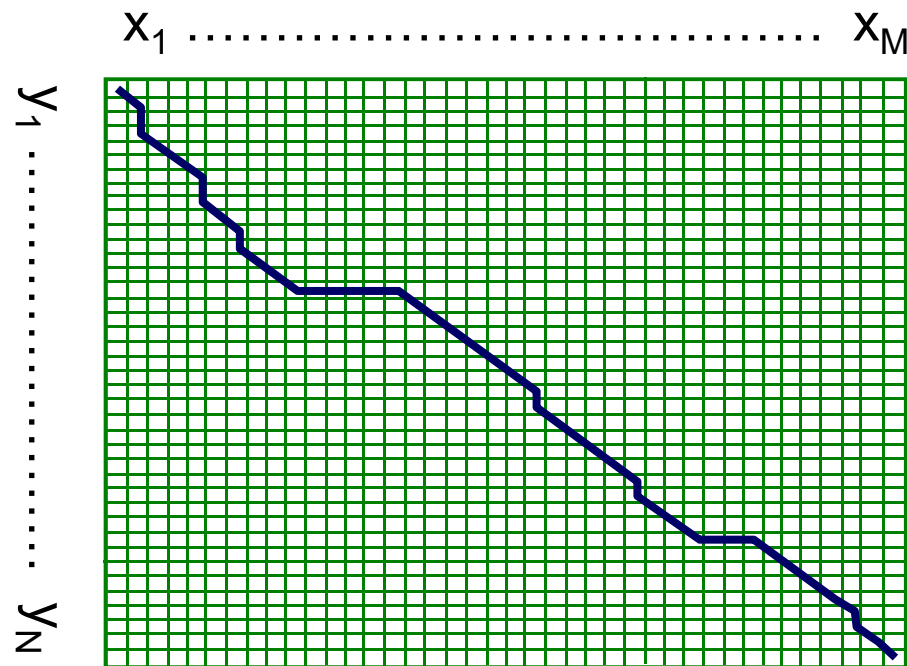
$$D(i, j) = \max \begin{cases} D(i-1, j) & - d \\ D(i, j-1) & - d \\ D(i-1, j-1) + s[x(i), y(j)] & (s: \text{score}) \end{cases}$$

- Término:

$D(N, M)$ is distance



A Matriz Needleman-Wunsch



(Note que a origem é no canto superior esquerdo.)



Uma Variante do Algoritmo Básico:

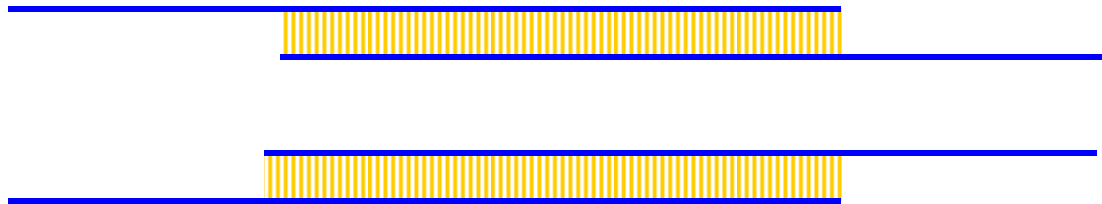
- Talvez esteja correto ter um número ilimitado de lacunas # no começo e no fim:

-----CTATCACCTGACCTCCAGGCCGATGCCCCTTCCGGC
GCGAGTTCATCTATCAC--GACCGC--GGTCG-----

- Sendo assim, nós não queremos penalizar lacunas nas extremidades

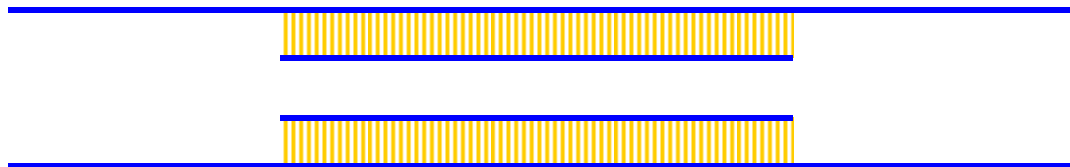


Diferentes tipos de Sobreposições (*Overlap*)



Exemplo:

2 sobreposições “lidas” de um projeto de sequenciamento

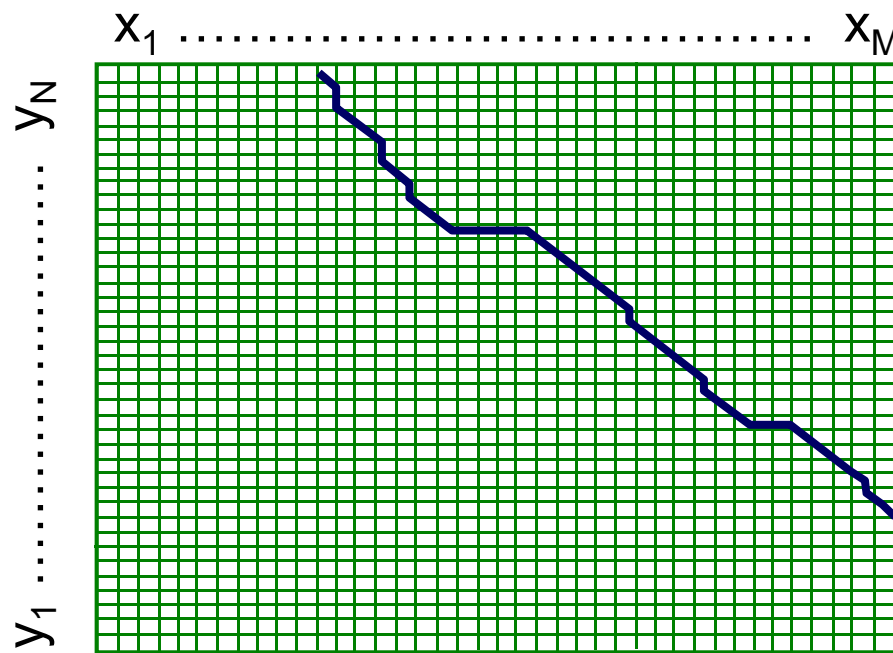


Exemplo:

Procurar um gene do rato dentro de um cromossomo humano



Variante de Detecção de Sobreposição



Mudanças:

1. Inicialização

For all i, j ,
 $F(i, 0) = 0$
 $F(0, j) = 0$

2. Término

$$F_{\text{OPT}} = \max \begin{cases} \max_i F(i, N) \\ \max_j F(M, j) \end{cases}$$



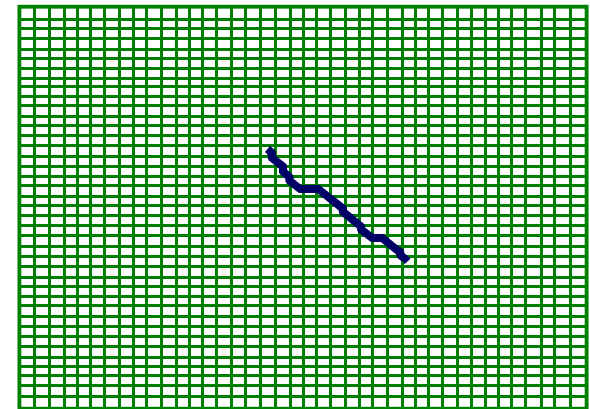
O Problema de Alinhamento Local

Dadas duas strings

$$x = x_1 \dots x_M, \quad y = y_1 \dots y_N$$

Encontrar substrings x' , y' as quais similaridade
(valor ótimo alinhamento global)
é máxima

$x = \text{aaaacc} \boxed{\text{ccgggg}} \text{tta}$
 $y = \text{tt} \boxed{\text{ccgggga}} \text{accaacc}$





O Algoritmo Smith-Waterman

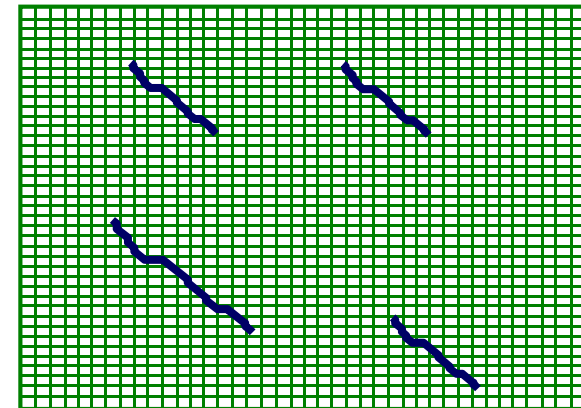
Ideia: Ignorar regiões de mal alinhamento

Modificações ao Needleman-Wunsch:

Inicialização:

$$F(0, j) = 0$$
$$F(i, 0) = 0$$

Iteração :

$$F(i, j) = \max \begin{cases} 0 \\ F(i - 1, j) - d \\ F(i, j - 1) - d \\ F(i - 1, j - 1) + s(x_i, y_j) \end{cases}$$




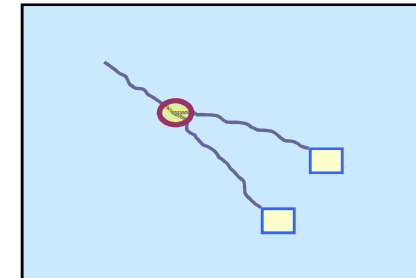
O Algoritmo Smith-Waterman

Término:

1. Se nós queremos o **melhor** alinhamento local...

$$F_{OPT} = \max_{i,j} F(i, j)$$

Encontrar F_{OPT} e fazer caminho de volta (trace back)



2. Se nós queremos **todos** os alinhamentos locais com **score > t**

Para todo i, j encontrar $F(i, j) > t$, e trace back?

Complicada pela sobreposição de alinhamentos locais



Exemplo de Alinhamento Local

X = ATCAT

Y = ATTATC

Seja:

$m = 1$ (1 ponto para combinações)

$d = 1$ (-1 ponto para del/ins/sub)

		A	T	T	A	T	C
	0	0	0	0	0	0	0
A	0						
T	0						
C	0						
A	0						
T	0						



Exemplo de Alinhamento Local

X = ATCAT
Y = ATTATC

		A	T	T	A	T	C
	0	0	0	0	0	0	0
A	0	1	0	0	1	0	0
T	0	0	2	1	0	2	0
C	0	0	1	1	0	1	3
A	0	1	0	0	2	1	2
T	0	0	2	0	1	3	2



Exemplo de Alinhamento Local

X = **ATCAT**

Y = **ATTATC**

		A	T	T	A	T	C
	0	0	0	0	0	0	0
A	0	1	0	0	1	0	0
T	0	0	2	1	0	2	0
C	0	0	1	1	0	1	3
A	0	1	0	0	2	1	2
T	0	0	2	0	1	3	2



Exemplo de Alinhamento Local

X = **ATC**AT

Y = ATT**ATC**

		A	T	T	A	T	C
	0	0	0	0	0	0	0
A	0	1	0	0	1	0	0
T	0	0	2	1	0	2	0
C	0	0	1	1	0	1	3
A	0	1	0	0	2	1	2
T	0	0	2	0	1	3	2



Distância Mínima de Edição

Distância Mínima de Edição em Computação Biológica



Processamento de Linguagem Natural

Modelagem de Linguagens

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Modelagem de Linguagens

Introdução a N-grams



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Modelos de Linguagem Probabilísticos

- Objetivo de Hoje: atribuir uma probabilidade a uma sentença
 - Tradução Automática:
 - » $P(\text{"high winds tonight"}) > P(\text{"large winds tonight"})$
 - Correção Ortográfica
 - » The office is about fifteen **minuets** from my house
 - $P(\text{"about fifteen minutes from"}) > P(\text{"about fifteen minuets from"})$
 - Reconhecimento de Discurso
 - » $P(\text{I saw a van}) \gg P(\text{eyes awe of an})$
 - + Sumarização, pergunta-resposta, etc., etc.!!

Por quê?



Modelagem de Linguagens Probabilísticas

- Objetivo: computar a probabilidade de uma sentença ou sequência de palavras:

$$P(W) = P(w_1, w_2, w_3, w_4, w_5, \dots, w_n)$$

- Tarefa Relacionada: probabilidade de uma próxima palavra:

$$P(w_5 | w_1, w_2, w_3, w_4)$$

- Um modelo que calcula qualquer uma destas:

$P(W)$ or $P(w_n | w_1, w_2, \dots, w_{n-1})$ é chamado de **Modelo de Linguagem**.

- Melhor: **Gramática** Mas **modelo de linguagem** ou **LM** é o padrão



Como calcular $P(W)$

- Como calcular esta probabilidade conjunta:
 - $P(\text{its, water, is, so, transparent, that})$
- Intuição: vamos contar com a Regra da Cadeia de Probabilidade



Lembrete: A Regra da Cadeia

- Probabilidade condicional de B dado A:

- $P(A|B) = P(A,B)/P(B) \rightarrow P(A,B) = P(A|B)P(B)$

- Mais variáveis:

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

- Regra da cadeia em geral

$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1, \dots, x_{n-1})$$



A Regra da Cadeia aplicada para calcular Probabilidade Conjunta de palavras em sentença

$$P(w_1 w_2 \dots w_n) = \prod_i P(w_i \mid w_1 w_2 \dots w_{i-1})$$

$P(\text{"its water is so transparent"}) =$

$P(\text{its}) \times P(\text{water} \mid \text{its}) \times P(\text{is} \mid \text{its water}) \times P(\text{so} \mid \text{its water is}) \times$
 $P(\text{transparent} \mid \text{its water is so})$



Como estimar estas probabilidades

- Poderíamos apenas contar e dividir?

$$P(\text{the} \mid \text{its water is so transparent that}) = \frac{\text{Count}(\text{its water is so transparent that the})}{\text{Count}(\text{its water is so transparent that})}$$

- Não! As frases possíveis são muitas!
- Nós nunca vamos ver dados suficiente para os estimar



Suposição de Markov

- Suposição simplificada:



Andrei Markov

$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{that})$

- Ou talvez

$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{transparent that})$



Suposição de Markov

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i \mid w_{i-k} \dots w_{i-1})$$

- Em outras palavras, nós aproximamos cada componente no produto

$$P(w_i \mid w_1 w_2 \dots w_{i-1}) \approx P(w_i \mid w_{i-k} \dots w_{i-1})$$



Caso mais simples: Modelo Unigram

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i)$$

Algumas sentenças automaticamente geradas de um modelo unigram

- fifth, an, of, futures, the, an, incorporated, a, a, the, inflation, most, dollars, quarter, in, is, mass
- thrift, did, eighty, said, hard, 'm, july, bullish
- that, or, limited, the



Modelo Bigram

- Condição da palavra anterior:

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-1})$$

- texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen
- outside, new, car, parking, lot, of, the, agreement, reached
- this, would, be, a, record, november



Modelos N-gram

- Podemos extender para trigrams, 4-grams, 5-grams
- Em geral este é um modelo de linguagem insuficiente
 - Porque linguagem tem **dependências de longa distância**:

“The **computer** which I had just put into the machine room on the fifth floor **crashed**.”
- Mas nós podemos sempre ir longe com modelos N-gram



Modelagem de Linguagens

Introdução a N-grams



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Modelagem de Linguagens

Estimar Probabilidades N-gram



Estimar Probabilidades Bigram

- Estimativa de Máxima Verossimilhança

$$P(w_i | w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

$$P(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$



Um Exemplo em Bigram

$$P(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

<s> I am Sam </s>

<s> Sam I am </s>

<s> I do not like green eggs and ham </s>

$$P(I | <s>) = \frac{2}{3} = .67$$

$$P(\text{Sam} | <s>) = \frac{1}{3} = .33$$

$$P(\text{am} | I) = \frac{2}{3} = .67$$

$$P(</s> | \text{Sam}) = \frac{1}{2} = 0.5$$

$$P(\text{Sam} | \text{am}) = \frac{1}{2} = .5$$

$$P(\text{do} | I) = \frac{1}{3} = .33$$



Mais exemplos: Sentenças do Projeto de Restaurante de Berkeley

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day



Contagem de Bigram Bruto

- Saída de 9222 sentenças

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0



Probabilidades de Bigram Bruto

$$P(A,B) = P(B|A)P(A)$$

- Normalizado por unigrams:

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

- Resultado:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0



Estimativas Bigram de Probabilidades de Sentença

$P(<s> \text{ I want english food } </s>) =$

$P(\text{I} | <s>)$

$\times P(\text{want} | \text{I})$

$\times P(\text{english} | \text{want})$

$\times P(\text{food} | \text{english})$

$\times P(</s> | \text{food})$

$= 0.25 \times 0.33 \times 0.011 \times 0.5 \times 0.68 = 0.000031$



Que Tipos de Conhecimento?

- $P(\text{english} | \text{want}) = .0011$
- $P(\text{chinese} | \text{want}) = .0065$
- $P(\text{to} | \text{want}) = .66$
- $P(\text{eat} | \text{to}) = .28$
- $P(\text{food} | \text{to}) = 0$
- $P(\text{want} | \text{spend}) = 0$
- $P(i | \langle s \rangle) = .25$



Pontos Práticos

- Nós fazemos tudo em espaço de log
 - Evitar underflow
 - (além do que adição é mais rápida que multiplicação)

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$



Toolkit de Modelagem de Linguagens

- SRILM

- <http://www.speech.sri.com/projects/srilm/>

- Em C++



Google N-Gram Release, Agosto de 2006

AUG

3

All Our N-gram are Belong to You

Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word **n-gram models** for a variety of R&D projects,

...

That's why we decided to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.

<https://books.google.com/ngrams>



Google N-Gram Release

- serve as the incoming 92
- serve as the incubator 99
- serve as the independent 794
- serve as the index 223
- serve as the indication 72
- serve as the indicator 120
- serve as the indicators 45
- serve as the indispensable 111
- serve as the indispensable 40
- serve as the individual 234

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



Modelagem de Linguagens

Estimar Probabilidades N-gram



Modelagem de Linguagens

Avaliação e Perplexidade



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Avaliação: Quão bom é o nosso modelo?

- Nosso modelo de linguagem prefere boas sentenças às ruins?
 - Atribui probabilidades mais altas à sentenças “reais” ou “frequentemente observadas”
 - Do que sentenças “não gramaticalmente bem formadas” ou “raramente observadas”?
- Treinamos parâmetros do nosso modelo por um **conjunto de treinamento**.
- Testamos a performance do modelo com dados não vistos anteriormente.
 - Um **conjunto de teste** é um conjunto de dados que é totalmente diferente do de treinamento, totalmente não utilizado.
 - Uma **métrica de avaliação** nos mostra o quão bom nosso modelo foi com o conjunto de testes.



Avaliação Extrínseca de Modelos N-gram

- Melhor avaliação para comparar modelos A e B
 - Coloque cada modelo em uma tarefa
 - Corretor Ortográfico, Reconhecedor de Discurso, Sistema MT
 - Executar a tarefa, obter uma precisão para A e B
 - Quantas palavras com erros ortográficos corrigidas corretamente
 - Quantas palavras traduzidas corretamente
 - Compare precisão de A e B



Dificuldade de Avaliação Extrínseca (in-vivo) de Modelos N-gram

- Avaliação Extrínseca
 - Demorado; pode levar dias ou semanas
- Então
 - As vezes utilizar avaliação **intrínseca: perplexidade**
 - Má aproximação
 - A menos que os dados de teste se pareçam com os dados de treinamento
 - Então, **geralmente só é útil em experimentos piloto**
 - Mas é útil para pensar.



Intuição de Perplexidade

- O Jogo de Shannon:
 - Quão bem podemos prever a próxima palavra?

I always order pizza with cheese and _____

The 33rd President of the US was _____

I saw a _____
 - Unigrams são terríveis neste jogo. (Por quê?)
- Um modelo melhor de um texto
 - é aquele que atribui uma probabilidade mais elevada para a palavra que ocorre efetivamente

mushrooms 0.1
pepperoni 0.1
anchovies 0.01
....
fried rice 0.0001
....
and 1e-100



Perplexidade

O melhor modelo de linguagem é aquele que melhor prediz uma base de teste nunca vista

- Apresenta a mais alta $P(\text{sentença})$

Perplexidade é a probabilidade inversa do conjunto de teste, normalizado pelo número de palavras:

Regra de Cadeia:

Para bigrams:

$$PP(W) = P(w_1 w_2 \dots w_N)^{\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_1 \dots w_{i-1})}}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_{i-1})}}$$

Minimizar perplexidade é o mesmo que maximizar probabilidade



O Jogo de Intuição de Shannon para Perplexidade

- De Josh Goodman
- Quanto difícil é a tarefa de reconhecer dígitos '0,1,2,3,4,5,6,7,8,9'
 - Perplexidade = 10
- Quanto difícil é reconhecer (30,000) nomes na Microsoft.
 - Perplexidade = 30,000
- Se um sistema tem que reconhecer
 - Operador (1 em 4)
 - Vendas (1 em 4)
 - Suporte Técnico (1 em 4)
 - 30,000 nomes (1 em 120,000 cada)
 - Perplexidade é $53 = [(1/4) \times (1/4) \times (1/4) \times (1/120,000)]^{(1/4)}$
- Perplexidade é um fator de ramificação equivalente ponderado



Perplexidade como Fator de Ramificação

- Vamos supor uma sentença que consiste de dígitos aleatórios
- Qual é a perplexidade de uma sentença de acordo com um modelo que atribui $P=1/10$ para cada dígito?

$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} \\ &= \left(\frac{1}{10}\right)^{-\frac{1}{N}} \\ &= \frac{1}{10}^{-1} \\ &= 10 \end{aligned}$$



Menor perplexidade = melhor modelo

- Treinamento 38 milhões de palavras e teste 1.5 milhões, WSJ

Ordenação N-gram	Unigram	Bigram	Trigram
Perplexity	962	170	109



Modelagem de Linguagens

Avaliação e Perplexidade



Modelagem de Linguagens

Generalização e Zeros



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



O Método de Visualização de Shannon

- Escolha um bigram aleatório ($\langle s \rangle, w$) de acordo com sua probabilidade
 - Agora escolha um bigram aleatório (w, x) de acordo com sua probabilidade
 - E assim por diante até que nós escolhemos $\langle /s \rangle$
 - E por fim junte as palavras
- $\langle s \rangle$ I
I want
want to
to eat
eat Chinese
Chinese food
food $\langle /s \rangle$
I want to eat Chinese food



Aproximação de Shakespeare

Unigram

To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
Every enter now severally so, let
Hill he late speaks; or! a more to leg less first you enter
Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

Bigram

What means, sir. I confess she? then all sorts, he is trim, captain.
Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

Trigram

Sweet prince, Falstaff shall die. Harry of Monmouth's grave.
This shall forbid it should be branded, if renown made it empty.
Indeed the duke; and had a very good friend.
Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

Quadrigram

King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;
Will you not tell me who I am?
It cannot be but so.
Indeed the short and the long. Marry, 'tis a noble Lepidus.



Shakespeare como Corpus

- $N=884,647$ tokens, $V=29,066$
- Shakespeare produziu 300,000 tipos de bigram de um total de $V^2= 844$ milhões possíveis bigrams.
 - Assim, 99,96% dos possíveis bigrams nunca foram vistos (tem entradas zero na tabela)
- Quadrigram pior: O que está sendo apresentado parece Shakespeare porque *é* Shakespeare



O Jornal de Wall Street não é Shakespeare (sem ofensa)

Unigram

Months the my and issue of year foreign new exchange's september were recession ex-
change new endorsed a acquire to six executives

Bigram

Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor
would seem to complete the major central planners one point five percent of U. S. E. has
already old M. X. corporation of living on information such as more frequently fishing to
keep her

Trigram

They also point to ninety nine point six billion dollars from two hundred four oh six three
percent of the rates of interest stores as Mexico and Brazil on market conditions



Os Perigos do Sobretreinamento (*Overfitting*)

- N-grams só funciona bem para predição de palavras se o corpo de teste se parece com o corpo de treino
 - Na vida real, isso geralmente não acontece
 - Nós precisamos treinar modelos robustos que generalizam!
 - Um tipo de generalização: Zeros!
 - Coisas que nunca ocorrem no conjunto de treino
 - Mas ocorrem no conjunto de teste



Zeros

- Conjunto de Treinamento:
 - ... denied the allegations
 - ... denied the reports
 - ... denied the claims
 - ... denied the request
- Conjunto de Teste:
 - ... denied the offer
 - ... denied the loan

$$P(\text{"offer"} \mid \text{denied the}) = 0$$



Bigrams de Probabilidade Zero

- Bigrams com zero probabilidade
 - significa que nós vamos atribuir probabilidade 0 ao conjunto de teste!
- E portanto nós não podemos calcular perplexidade (não podemos dividir por 0)!



Modelagem de Linguagens

Generalização e Zeros



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Processamento de Linguagem Natural

Modelagem de Linguagens

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Modelagem de Linguagens

Suavização: Add-one ou (Laplace) Smoothing



A Intuição de Suavização (por Dan Klein)

- Quando temos estatísticas esparsas:

$P(w \mid \text{denied the})$

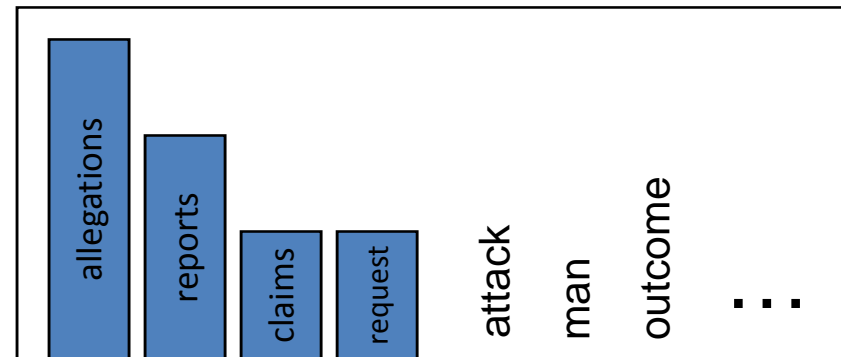
3 allegations

2 reports

1 claims

1 request

7 total



- Roubar massa de probabilidade para generalizar melhor

$P(w \mid \text{denied the})$

2.5 allegations

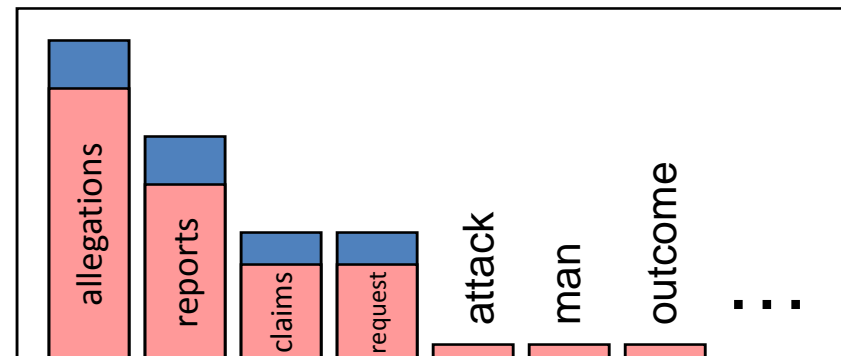
1.5 reports

0.5 claims

0.5 request

2 other

7 total





Estimação Add-one

- Também chamada de suavização de Laplace (smoothing)
- “Finge” que nós vimos cada palavra uma vez a mais do que realmente vimos
- Apenas adicione um a todas as contagens!

- Estimativa MLE:

$$P_{MLE}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i)}{\alpha(w_{i-1})}$$

- Estimativa Add-1:

$$P_{Add-1}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + 1}{\alpha(w_{i-1}) + V}$$



Estimativas de Máxima Verossimilhança

- A estimativa de máxima verossimilhança (MLE)
 - de alguns parâmetros de um modelo M de um conjunto de treinamento T
 - maximiza a verossimilhança do conjunto de treinamento T dado o modelo M
- Suponha que a palavra “bagel” aparece 400 vezes em um montante de um milhão de palavras
- Qual é a probabilidade de que uma palavra aleatória de algum outro texto será “bagel”?
- A estimativa MLE é $400/1,000,000 = .0004$
- Esta pode ser uma má estimativa para algum outro corpo de texto
 - Mas esta é a **estimativa** que torna **mais provável** que “bagel” ocorra 400 vezes em um texto de um milhão de palavras.



Textos do Restaurante de Berkeley: Contagens Bigram com Suavização de Laplace

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1



Bigrams com Suavização Laplace

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058



Contagens Reconstituídas

$$c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n) + 1] \times C(w_{n-1})}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16



Comparação Original x Nova

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16



Estimação Add-1 é um instrumento cego

- Assim add-1 não é usado para N-grams:
 - Nós veremos métodos melhores
- Mas add-1 é usado para suavizar outros modelos PLN
 - Para Classificação de Texto
 - Em domínios onde o número de zeros não é tão grande.



Modelagem de Linguagens

Suavização: Add-one ou (Laplace) Smoothing



Modelagem de Linguagens

Interpolação, Recuo e
LMs de Escala Web



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Recuo e Interpolação

- As vezes ajuda a usar **menos** contexto
 - Condição em menos contexto para contextos em que não se aprendeu muito sobre
- **Recuo:**
 - Usar trigram se você tiver boas evidências,
 - Se não, usar bigram ou unigram
- **Interpolação:**
 - Misturar unigram, bigram, trigram
- Interpolação funciona melhor



Interpolação Linear

- Interpolação Simples

$$\begin{aligned}\hat{P}(w_n|w_{n-1}w_{n-2}) = & \lambda_1 P(w_n|w_{n-1}w_{n-2}) \\ & + \lambda_2 P(w_n|w_{n-1}) \\ & + \lambda_3 P(w_n)\end{aligned}\quad \sum_i \lambda_i = 1$$

- Lambdas condicionais no contexto:

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) = & \lambda_1 (w_{n-2}^{n-1}) P(w_n|w_{n-2}w_{n-1}) \\ & + \lambda_2 (w_{n-1}^{n-1}) P(w_n|w_{n-1}) \\ & + \lambda_3 (w_{n-2}^{n-1}) P(w_n)\end{aligned}$$



Como escolher os lambdas?

- Utilizar um conjunto de **resistência (held-out)**



- Escolher λ s para maximizar a probabilidade dos dados held-out:
 - Encontrar as probabilidades N-gram (nos dados de treinamento)
 - Posteriormente, procurar por λ s que dão a maior probabilidade para o conjunto de held-out:

$$\log P(w_1 \dots w_n | M(\lambda_1 \dots \lambda_k)) = \sum_i \log P_{M(\lambda_1 \dots \lambda_k)}(w_i | w_{i-1})$$



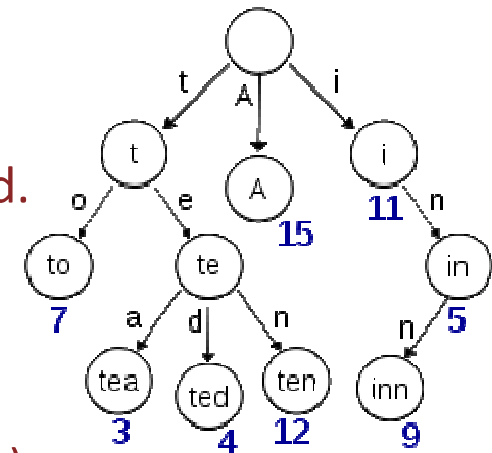
Palavras desconhecidas: Tarefas de vocabulário Abertas x Fechadas

- Se sabemos todas as palavras com antecedência
 - Vocabulário V é fixado
 - Tarefa de Vocabulário Fechada
- Sempre que não sabemos
 - **Out Of Vocabulary (Fora do vocabulário)** = palavras OOV
 - Tarefa de Vocabulário Aberta
- Ou então: criar um token de palavra desconhecida <UNK>
 - Obtenção de probabilidades de <UNK> (Treinamento)
 - Criar um vocabulário léxico L de tamanho V
 - Na fase de normalização do texto, qualquer palavra que não está em L muda para <UNK>
 - Agora nós obtemos (através de treinamento) suas probabilidades como uma palavra normal
 - No momento de decodificação
 - Se entrada de texto: Usar probabilidades de UNK para qualquer palavra que não esteja em treinamento



N-grams de Escala Web (Alta Escala)

- Como lidar com, e.g., Coleção Google de N-gram
- Poda
 - Apenas armazenar N-grams com contagem > threshold.
 - Remover singletons de n-grams de maior ordem
 - Poda baseada em Entropia
- Eficiência
 - Estrutura de dados eficientes como árvores radix (tries)
 - Filtros Bloom: aproximar modelos de linguagem
 - Armazenar palavras como índices, não strings
 - Usar codificação Huffman para encaixar um grande número de palavras em dois bytes
 - Quantificar probabilidades (4-8 bits ao invés de 8-byte float)





Suavização para N-grams de Escala Web

- “Recuo estúpido (Stupid backoff)” (Brants *et al.* 2007)
- Não ligue, apenas use frequências relativas

$$\mathcal{S}(w_i | w_{i-k+1}^{j-1}) = \begin{cases} \frac{\text{count}(w_{i-k+1}^j)}{\text{count}(w_{i-k+1}^{j-1})} & \text{if } \text{count}(w_{i-k+1}^j) > 0 \\ 0.4 \mathcal{S}(w_i | w_{i-k+2}^{j-1}) & \text{otherwise} \end{cases}$$

$$\mathcal{S}(w_i) = \frac{\text{count}(w_i)}{N}$$



Resumo: Suavização N-gram

- Suavização Add-1:
 - Bom para categorização de texto, não para modelagem de linguagens
- O método mais comumente utilizado:
 - Kneser-Ney Estendido e Interpolado
- Para N-grams muito grandes como os Web:
 - Stupid backoff



Modelagem de Linguagem Avançada

- Modelos Discriminativos:
 - Escolher pesos n-gram para melhorar uma tarefa, não para ajustar o conjunto de treinamento
- Modelos Baseados em *Parsing*
- Modelos de *Caching*
 - Palavras usadas recentemente são mais prováveis de aparecerem

$$P_{CACHE}(w | history) = \lambda P(w_i | w_{i-2} w_{i-1}) + (1 - \lambda) \frac{\alpha(w \in history)}{|history|}$$

- Tem desempenho pobre para reconhecimento de discurso (Por quê?)



Modelagem de Linguagens

Interpolação, Recuo e
LMs de Escala Web



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Modelagem de Linguagens

Avançado:
Suavização Good-Turing



Lembrete: Suavização Add-1 (Laplace)

$$P_{Add-1}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + 1}{\alpha(w_{i-1}) + V}$$



Formulações mais gerais: Add-k

$$P_{Add-k}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + k}{\alpha(w_{i-1}) + kV}$$

$$P_{Add-k}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + m(\frac{1}{V})}{\alpha(w_{i-1}) + m}$$



Suavização Prévia Unigram

$$P_{Add-k}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + m(\frac{1}{V})}{\alpha(w_{i-1}) + m}$$

$$P_{\text{UnigramPrior}}(w_i | w_{i-1}) = \frac{\alpha(w_{i-1}, w_i) + mP(w_i)}{\alpha(w_{i-1}) + m}$$



Algoritmos Avançados de Suavização

- Linha de pensamento usada por muitos algoritmos de suavização
 - Good-Turing
 - Kneser-Ney
 - Witten-Bell
- Use a contagem de coisas que vimos **uma vez**
 - para ajudar a estimar a contagem de coisas que **nunca vimos**



Notação: N_c = Contagem de frequência de c

- N_c = contagem de coisas que vimos c vezes
- Sam I am I am Sam I do not eat

I 3

sam 2

am 2

do 1

not 1

eat 1

$$N_1 = 3$$

$$N_2 = 2$$

$$N_3 = 1$$



Intuição da Suavização Good-Turing

- Você está pescando (um cenário de Josh Goodman) e pega:
 - 10 carpas, 3 percas, 2 sardinhas, 1 truta, 1 salmão, 1 enguia = 18 peixes
- Qual a probabilidade da próxima espécie ser truta?
 - $1/18$
- Quão provável é a próxima ser uma nova espécie (i.e. bagre ou baiacu)
 - Vamos usar nossa estimativa de coisas-que-vi-uma-vez para estimar novas coisas.
 - $3/18$ (Sendo, $N_1=3$)
- Considerando isso, quão provável é a nova espécie ser truta?
 - Precisa ser menos de $1/18$
 - Como estimar?



Cálculos Good Turing

$$P_{GT}^* (\text{coisas com frequência zero}) = \frac{N_1}{N} \quad c^* = \frac{(c+1)N_{c+1}}{N_c}$$

- Não visto (baiacu ou bagre)
 - $c = 0$:
 - MLE $p = 0/18 = 0$
 - $P_{GT}^* (\text{Não visto}) = N_1/N = 3/18$
- Visto uma vez (truta)
 - $c = 1$
 - MLE $p = 1/18$
 - $C^*(\text{truta}) = 2 * N_2/N_1$
 $= 2 * 1/3$
 $= 2/3$
 - $P_{GT}^*(\text{truta}) = (2/3)/18 = 1/27$



Intuição de Good-Turing - Ney et al.

H. Ney, U. Essen, and R. Kneser, 1995. On the estimation of 'small' probabilities by leaving-one-out.
IEEE Trans. PAMI. 17:12,1202-1212



Palavras Held-out:

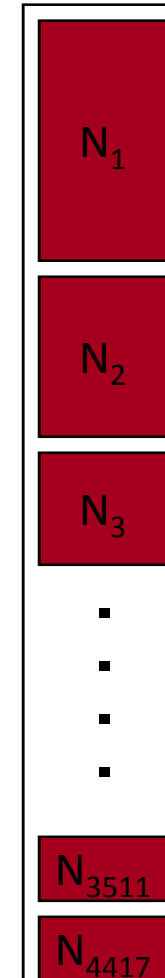


Intuição de Good-Turing - Ney et al. (slide de Dan Klein)

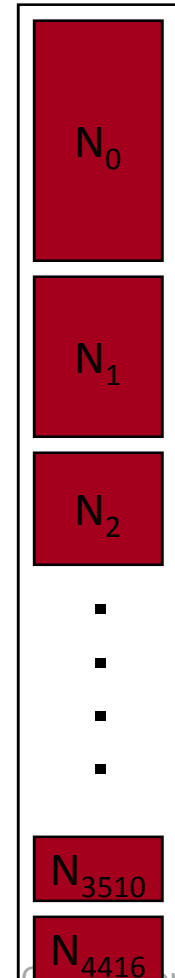
- Intuição da validação deixe-um-fora
 - Tire cada uma das c palavras de treinamento por turno
 - c conjuntos de treinamento de tamanho $c-1$, held-out de tamanho 1
 - Que fração de palavras held-out não são vistas em treinamento?
 - N_1/c
 - Que fração de palavras held-out são vistas k vezes em treinamento?
 - $(k+1)N_{k+1}/c$
 - Assim, no futuro, esperamos $(k+1)N_{k+1}/c$ das palavras serem aquelas com contagem de treinamento k
 - Existem N_k palavras com contagem de treinamento k
 - Cada um deve ocorrer com probabilidade:
 - $(k+1)N_{k+1}/c/N_k$
 - ...ou contagem esperada:

$$k^* = \frac{(k+1)N_{k+1}}{N_k}$$

Training



Held out



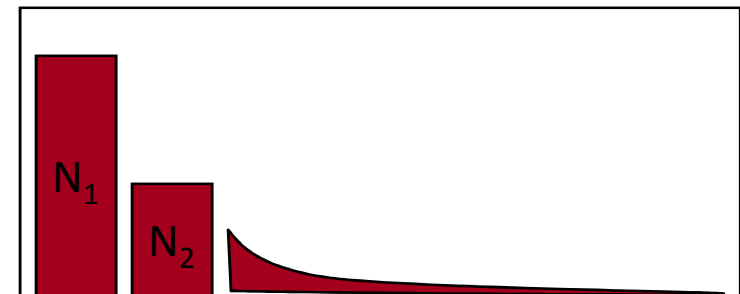
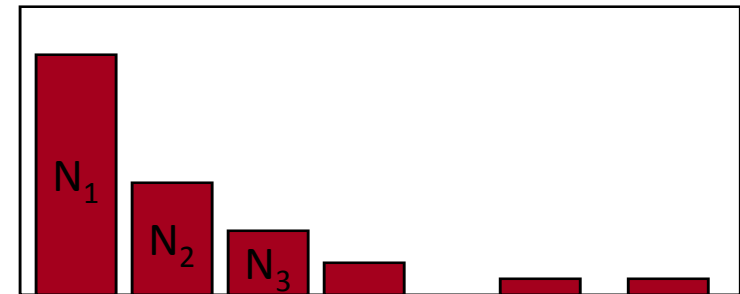
On: dtpc.br



Complicações de Good-Turing

(slide de Dan Klein)

- Problema:
 - A equação $k^* = \frac{(k+1)N_{k+1}}{N_k}$ é indefinida para $N_k = 0$
 - Simple Good-Turing [Gale e Sampson]: substituir N_k empiricamente com uma lei de potência de melhor ajuste, uma vez que contagens se tornaram não confiáveis.
 - Interporla N_k em função de (N_{k+1}, N_{k-1})





Números Resultantes de Good-Turing

- Números por Church e Gale (1991)
- 22 milhões de palavras do AP Newswire

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

Contagem c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25



Modelagem de Linguagens

Avançado:
Suavização Good-Turing



Modelagem de Linguagens

Avançado:
Suavização Kneser-Ney



Números Resultantes de Good-Turing

- Números por Church e Gale(1991)
- 22 milhões de palavras do AP Newswire

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

- Na tabela ao lado, parece que (tirando 0 e 1):

$$c^* = c - 0,75$$

Contagem c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25



Interpolação Absoluta com Desconto

- Salve-nos algum tempo e apenas subtraia 0.75 (ou algum d)!

$$P_{\text{AbsoluteDiscounting}}(w_i \mid w_{i-1}) = \frac{\overset{\text{Bigram descontado}}{c(w_{i-1}, w_i) - d}}{c(w_{i-1})} + \overset{\text{Peso de interpolação}}{\lambda(w_{i-1})} \overset{\text{unigram}}{P(w)}$$

- (Talvez mantendo alguns valores extras de d para contagens 1 e 2)
- Mas devemos realmente só usar o unigram regular $P(w)$?



Suavização I Kneser-Ney

- Melhor estimador para probabilidades de unigrams de baixa ordem!
 - Shannon game: *I can't see without my reading Frglasses?*
 - “Francisco” é mais comum que “glasses”
 - ... mas “Francisco” sempre vem depois de “San”
- O unigram é útil exatamente quando nós ainda não vimos esse bigram!
- Ao invés de $P(w)$: “O quão provável é w ”
- $P_{\text{continuation}}(w)$: “O quão provável é w aparecer em um novo contexto?”
 - Para cada palavra, contar o número de tipos de bigram que ela completa
 - Cada tipo de bigram era um novo contexto* na primeira vez que foi visto

$$P_{\text{CONTINUATION}}(w) \propto |\{w_{i-1} : \alpha(w_{i-1}, w) > 0\}|$$



Sauvização II Kneser-Ney

- Quantas vezes w aparece como uma nova continuação:

$$P_{CONTINUATION}(w) \propto |\{w_{i-1} : \alpha(w_{i-1}, w) > 0\}|$$

- Normalizado pelo número total de tipos de bigram de palavra

$$|\{(w_{j-1}, w_j) : \alpha(w_{j-1}, w_j) > 0\}|$$

$$P_{CONTINUATION}(w) = \frac{|\{w_{i-1} : \alpha(w_{i-1}, w) > 0\}|}{|\{(w_{j-1}, w_j) : \alpha(w_{j-1}, w_j) > 0\}|}$$



Suavização III Kneser-Ney

- Metáfora Alternativa: O número de tipos de palavras vistas precedendo w

$$| \{ w_{i-1} : \alpha(w_{i-1}, w) > 0 \} |$$

- normalizado pelo numero de palavras precedendo todas as palavras:

$$P_{CONTINUATION}(w) = \frac{| \{ w_{i-1} : \alpha(w_{i-1}, w) > 0 \} |}{\sum_{w'} | \{ w'_{i-1} : \alpha(w'_{i-1}, w') > 0 \} |}$$

- Uma palavra frequente (Francisco) ocorrendo em um único contexto (San) vai ter uma probabilidade baixa de continuação



Suavização IV Kneser-Ney

$$P_{KN}(w_i | w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{CONTINUATION}(w_i)$$

λ é uma constante de normalização; a massa de probabilidade que nós descontamos

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} |\{w : c(w_{i-1}, w) > 0\}|$$

O desconto normalizado

Número de tipos de palavra que seguem w_{i-1}
= # de tipos de palavras que descontamos
= # de vezes que aplicamos desconto normalizado



Modelagem de Linguagens

Avançado:
Suavização Kneser-Ney



Processamento de Linguagem Natural

Correção Ortográfica

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Correção Ortográfica e Canal com Ruído

A Tarefa de Correção Ortográfica

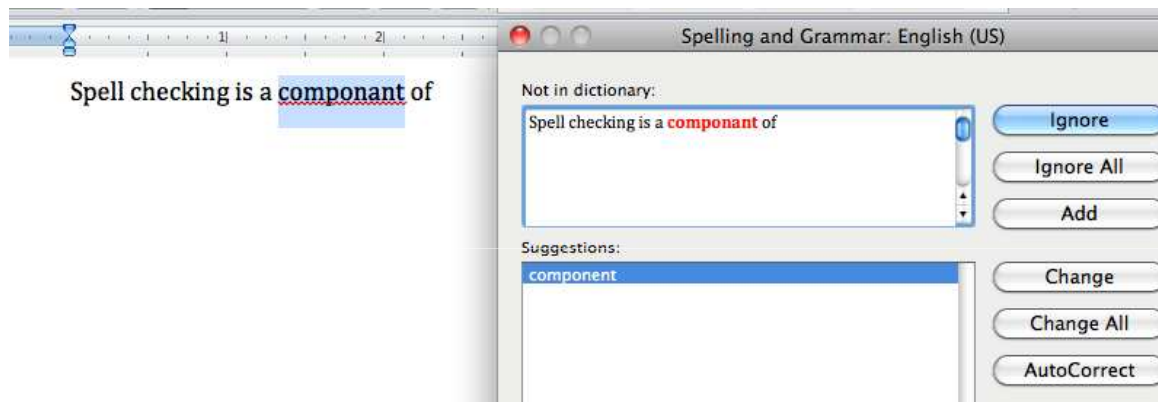


UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

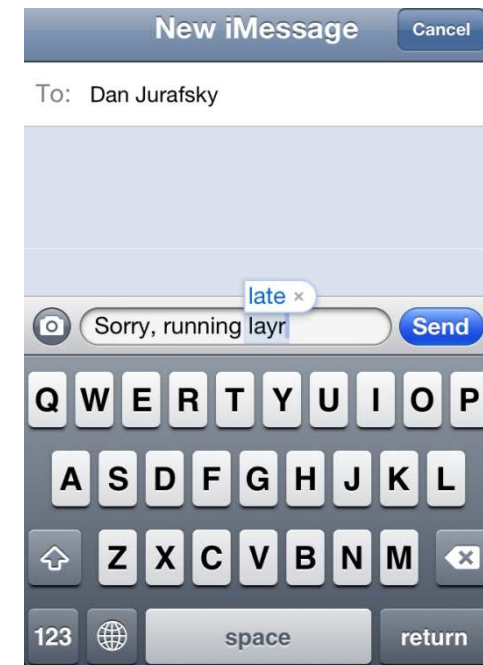


Aplicações para Correção Ortográfica

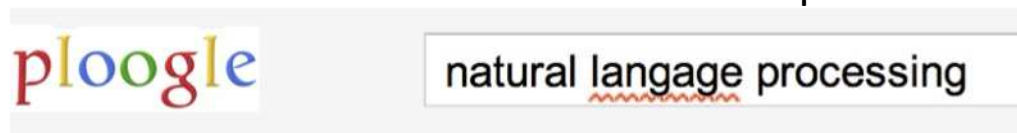
Processamento de Textos



Telefones



Pesquisa na Internet



Showing results for [natural language processing](#)
Search instead for [natural language processing](#)



Tarefas de Ortografia

- Detecção de Erros de Ortografia
- Correção de Erros de Ortografia:
 - Correção automática
 - hte → the
 - Sugestão de correção
 - Listas de sugestões



Tipos de Erros de Ortografia

- Erros Non-word
 - *graffe* → *giraffe*
- Erros Real-word
 - Erros Tipográficos
 - *three* → *there*
 - Erros Cognitivos (homófono)
 - *piece* → *peace*,
 - *too* → *two*



Taxas de Erros Ortográficos

26%: Consultas na Web *Wang et al. 2003*

13%: Redigitação, sem backspace *Whitelaw et al. English&German*

7%: Palavras corrigidas redigitando em agendas eletrônicas

2%: Palavras não corrigidas em agendas eletrônicas *Soukoreff &MacKenzie 2003*

1-2%: Redigitação: *Kane and Wobbrock 2007, Gruden et al. 1983*



Erros de Ortografia Non-word

- Detecção de erros ortográficos Non-word:
 - Qualquer palavra que não esteja no **dicionário** é um erro
 - Quanto maior o dicionário melhor
- Correção de erros ortográficos Non-word:
 - Geração de **candidatos**: palavras reais que são similares ao erro
 - Escolher o que é melhor:
 - Menor distância de edição ponderada
 - Maior probabilidade de canal com ruído



Erros de Ortografia Real Word

- Para cada palavra w , gerar um conjunto de candidatos:
 - Encontrar palavras candidatas com ***pronúncia*** similar
 - Encontrar palavras candidatas com ***ortografia*** similar
 - Incluir w no conjunto de candidatos
- Escolher o melhor candidato
 - Canal com Ruído
 - Classificador



Correção Ortográfica e Canal com Ruído

A Tarefa de Correção Ortográfica



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

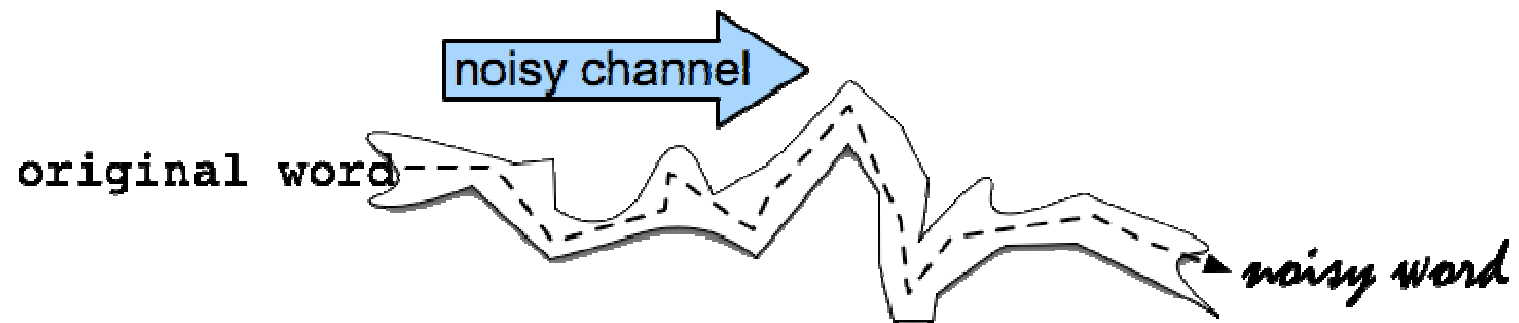


Correção Ortográfica e Canal com Ruído

Ortografia: Modelo de Canal com
Ruído (Noisy Channel)



Canal com Ruído (Noisy Channel)





Canal com Ruído (Noisy Channel)

- Vemos uma observação x de uma palavra mal escrita
- Encontrar a palavra correta w

$$\begin{aligned}\hat{w} &= \operatorname{argmax}_{w \in V} P(w | x) \\ &= \operatorname{argmax}_{w \in V} \frac{P(x | w) P(w)}{P(x)} \\ &= \operatorname{argmax}_{w \in V} P(x | w) P(w)\end{aligned}$$



História: Noisy Channel para Ortografia proposto por volta de 1990

- **IBM**

- Mays, Eric, Fred J. Damerau and Robert L. Mercer. 1991. Context based spelling correction. *Information Processing and Management*, 23(5), 517–522

- **AT&T Bell Labs**

- Kernighan, Mark D., Kenneth W. Church, and William A. Gale. 1990. A spelling correction program based on a noisy channel model. *Proceedings of COLING 1990*, 205-210



Exemplo de Erro Ortográfico Non-word

acress



Geração de Candidato

- Palavras com ortografia similar
 - Distância pequena de edição ao erro
- Palavras com pronuncia similar
 - Distância pequena de edição de pronuncia ao erro



Distância de Edição Damerau-Levenshtein

- Distância mínima de edição entre duas strings, em que edições são:
 - Inserção
 - Remoção
 - Substituição
 - Transposição de duas letras adjacentes



Candidatos para `acress` com distância 1

Erro	Candidato de Correção	Letra Correta	Letra Errada	Tipo
<code>acress</code>	<code>actress</code>	<code>t</code>	<code>-</code>	remoção
<code>acress</code>	<code>cress</code>	<code>-</code>	<code>a</code>	inserção
<code>acress</code>	<code>caress</code>	<code>ca</code>	<code>ac</code>	transposição
<code>acress</code>	<code>access</code>	<code>c</code>	<code>r</code>	substituição
<code>acress</code>	<code>across</code>	<code>o</code>	<code>e</code>	substituição
<code>acress</code>	<code>acres</code>	<code>-</code>	<code>s</code>	inserção
<code>acress</code>	<code>acres</code>	<code>-</code>	<code>s</code>	inserção



Geração de Candidatos

- 80% dos erros tem distância de edição 1
- Quase todos os erros tem distância de edição 2
- Também permite inserção de **espaço** ou **hífen**
 - thisidea → this idea
 - inlaw → in-law



Modelo de Linguagem

- Usar qualquer um dos algoritmos de modelo de linguagem que vimos
- Unigram, bigram, trigram
- Correção Ortográfica Web-scale
 - Stupid backoff



Unigram: Probabilidade Prévia

Contagens de 404,253,213 palavras no Corpus of Contemporary English (COCA)

Palavra	Frequência da Palavra	P(palavra)
actress	9 , 321	.0000230573
cress	220	.0000005442
caress	686	.0000016969
access	37 , 038	.0000916207
across	120 , 844	.0002989314
acres	12 , 874	.0000318463



Probabilidade de Modelo de Canal

- Error model probability, Edit probability
- *Kernighan, Church, Gale 1990*
- *Palavra incorreta $x = x_1, x_2, x_3 \dots x_m$*
- *Palavra correta $w = w_1, w_2, w_3, \dots, w_n$*
- $P(x|w)$ = probabilidade de edição
 - (remoção/inserção/substituição/transposição)



Calcular probabilidade de erro: matriz de confusão

`del[x,y]:` contar (xy digitado como x)
`ins[x,y]:` contar (x digitado como xy)
`sub[x,y]:` contar (x digitado como y)
`trans[x,y]:` contar (xy digitado como yx)

Inserção e remoção condicionada ao caractere anterior (poderia ser em relação ao posterior também).

Matriz de Confusão para Erros de Ortografia

sub[X, Y] = Substitution of X (incorrect) for Y (correct)

X	Y (correct)																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0
b	0	0	9	9	2	2	3	1	0	0	0	5	11	5	0	10	0	0	2	1	0	0	8	0	0	0
c	6	5	0	16	0	9	5	0	0	0	1	0	7	9	1	10	2	5	39	40	1	3	7	1	1	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	3	0	1	0	43	30	22	0	0	4	0	2	0
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0
f	0	15	0	3	1	0	5	2	0	0	0	3	4	1	0	0	0	6	4	12	0	0	2	0	0	0
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0
i	103	0	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0
j	0	1	1	9	0	0	1	0	0	0	0	2	1	0	0	0	0	0	5	0	0	0	0	0	0	0
k	1	2	8	4	1	1	2	5	0	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	3
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	11	10	2	0	0	0	0	0	0
m	1	3	7	8	0	2	0	6	0	0	4	4	0	180	0	6	0	0	9	15	13	3	2	2	3	0
n	2	7	6	5	3	0	1	19	1	0	4	35	78	0	0	7	0	28	5	7	0	0	1	2	0	2
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	4	14	39	0	0	0	18	0
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0
q	0	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	0	15	0	0	5	3	20	1
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6
u	20	0	0	0	44	0	0	0	64	0	0	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0
v	0	0	7	0	0	3	0	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0
w	2	2	1	0	1	0	0	2	0	0	1	0	0	0	0	7	0	6	3	3	1	0	0	0	0	0
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0
y	0	0	2	0	15	0	1	7	15	0	0	0	2	0	6	1	0	7	36	8	5	0	0	1	0	0
z	0	0	0	7	0	0	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0



Geração da matriz de confusão

- [Lista de erros de Peter Norvig](#)
- [Lista de contagens de erros de edição única de Peter Norvig](#)

<http://norvig.com/ngrams/spell-errors.txt>



Modelo de Canal

Kernighan, Church, Gale 1990

$$P(x|w) = \begin{cases} \frac{\text{del}[w_{i-1}, w_i]}{\text{count}[w_{i-1} w_i]}, & \text{if deletion} \\ \frac{\text{ins}[w_{i-1}, x_i]}{\text{count}[w_{i-1}]}, & \text{if insertion} \\ \frac{\text{sub}[x_i, w_i]}{\text{count}[w_i]}, & \text{if substitution} \\ \frac{\text{trans}[w_i, w_{i+1}]}{\text{count}[w_i w_{i+1}]}, & \text{if transposition} \end{cases}$$



Modelo de Canal para across

Candidato de correção	Letra Correta	Letra Errada	$x w$	$P(x word)$
actress	t	-	c ct	.000117
cress	-	a	a #	.00000144
caress	ca	ac	ac ca	.00000164
access	c	r	r c	.000000209
across	o	e	e o	.0000093
acres	-	s	es e	.0000321
acres	-	s	ss s	.0000342



Probabilidade de Noisy Channel para acress

Candidato de correção	Letra Correta	Letra Errada	$x w$	$P(x word)$	$P(word)$	$10^9 * P(x w)P(w)$
actress	t	-	c ct	.000117	.0000231	2.7
cress	-	a	a #	.00000144	.000000544	.00078
caress	ca	ac	ac ca	.00000164	.00000170	.0028
access	c	r	r c	.000000209	.0000916	.019
across	o	e	e o	.00000093	.000299	2.8
acres	-	s	es e	.0000321	.0000318	1.0
acres	-	s	ss s	.0000342	.0000318	1.0



Probabilidade de Noisy Channel para acress

Candidato de correção	Letra Correta	Letra Errada	$x w$	$P(x word)$	$P(word)$	$10^9 * P(x w)P(w)$
actress	t	-	c ct	.000117	.0000231	2.7
cress	-	a	a #	.00000144	.000000544	.00078
caress	ca	ac	ac ca	.00000164	.00000170	.0028
access	c	r	r c	.000000209	.0000916	.019
across	o	e	e o	.0000093	.000299	2.8
acres	-	s	es e	.0000321	.0000318	1.0
acres	-	s	ss s	.0000342	.0000318	1.0



Usando um Modelo de Linguagem Bigram

- "a stellar and versatile **actress** whose combination of sass and glamour..."
- Contagens do *Corpus of Contemporary American English* com suavização add-1
- $P(\text{actress}|\text{versatile}) = .000021$ $P(\text{whose}|\text{actress}) = .0010$
- $P(\text{across}|\text{versatile}) = .000021$ $P(\text{whose}|\text{across}) = .000006$
- $P(\text{"versatile actress whose"}) = .000021 * .0010 = 210 \times 10^{-10}$
- $P(\text{"versatile across whose"}) = .000021 * .000006 = 1 \times 10^{-10}$



Usando um Modelo de Linguagem Bigram

- "a stellar and versatile **actress** whose combination of sass and glamour..."
- Contagens do *Corpus of Contemporary American English* com suavização add-1
- $P(\text{actress}|\text{versatile}) = .000021$ $P(\text{whose}|\text{actress}) = .0010$
- $P(\text{across}|\text{versatile}) = .000021$ $P(\text{whose}|\text{across}) = .000006$
- $P(\text{"versatile actress whose"}) = .000021 * .0010 = 210 \times 10^{-10}$
- $P(\text{"versatile across whose"}) = .000021 * .000006 = 1 \times 10^{-10}$



Avaliação

- Alguns conjuntos de teste de erros de ortografia
 - [Lista de erros comuns em Inglês da Wikipedia](#)
 - [Versão filtrada da lista Aspell](#)
 - [Conjunto de Erros Ortográficos de Birkbeck](#)
 - [Lista de Erros de Peter Norvig \(inclui Wikipedia e Birkbeck, para treinamento e teste\)](#)



Correção Ortográfica e Canal com Ruído

Ortografia: Modelo de Canal com
Ruído (Noisy Channel)



Correção Ortográfica e Canal com Ruído

Correção Ortográfica Real-Word



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Erros Ortográficos Real-word

- ...leaving in about fifteen **minuets** to go to her house.
 - The design **an** construction of the system...
 - Can they **lave** him my messages?
 - The study was conducted mainly **be** John Black.
-
- 25-40% dos erros ortográficos são palavras reais [Kukich 1992](#)



Resolvendo Erros Ortográficos Real-world

- Para cada palavra em uma sentença
 - Gerar um *conjunto de candidatos*
 - A palavra em si
 - Todas edições de letra única que são palavras em Inglês
 - Palavras que são homófonas
- Escolher melhores candidatos
 - Modelo Noisy channel
 - Classificador para tarefa específica

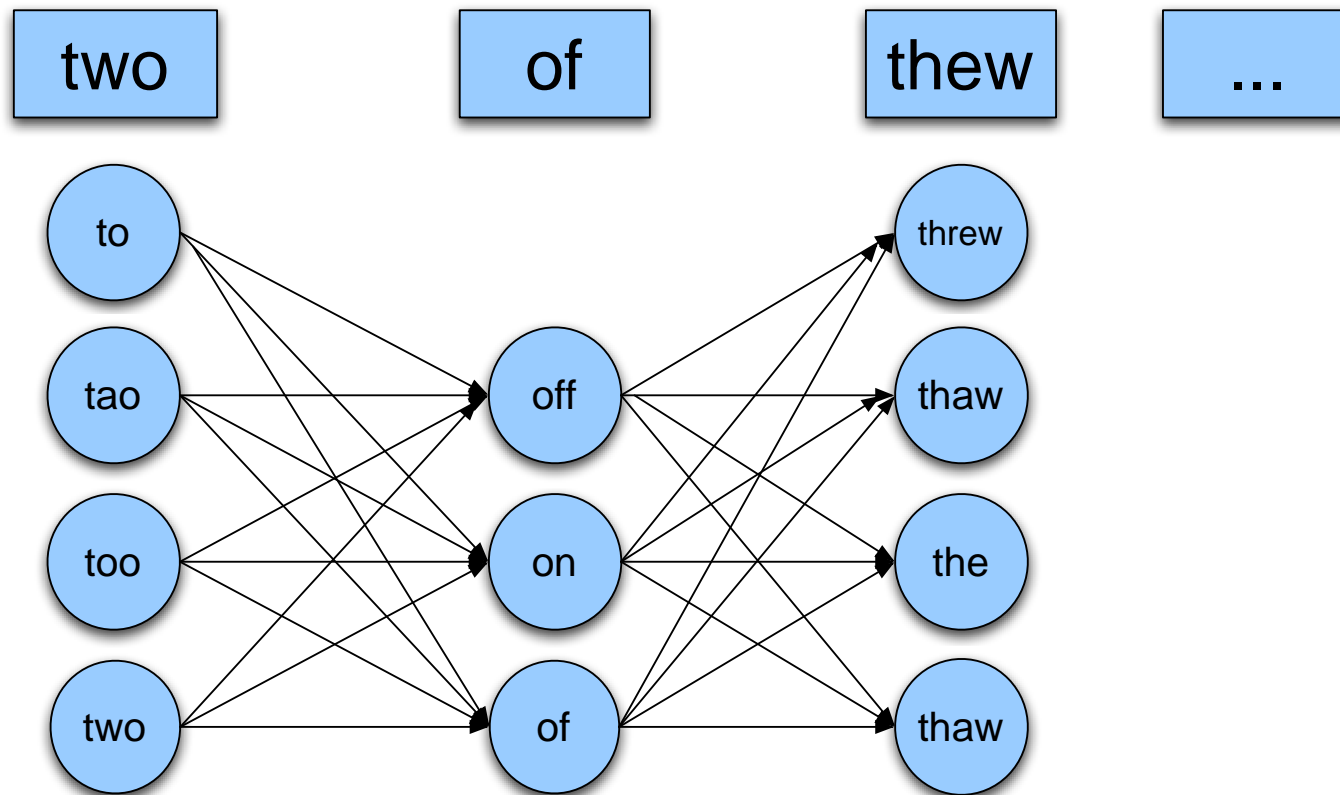


Noisy Channel para Correção Ortográfica em Real-Word

- Dada a sentença $W = w_1, w_2, w_3, \dots, w_n$
- Gerar um conjunto de candidatos para cada palavra w_i
 - $\text{Candidate}(w_1) = \{w_1, w'_1, w''_1, w'''_1, \dots\}$
 - $\text{Candidate}(w_2) = \{w_2, w'_2, w''_2, w'''_2, \dots\}$
 - $\text{Candidate}(w_n) = \{w_n, w'_n, w''_n, w'''_n, \dots\}$
- Escolher a sequência W que maximiza $P(W)$

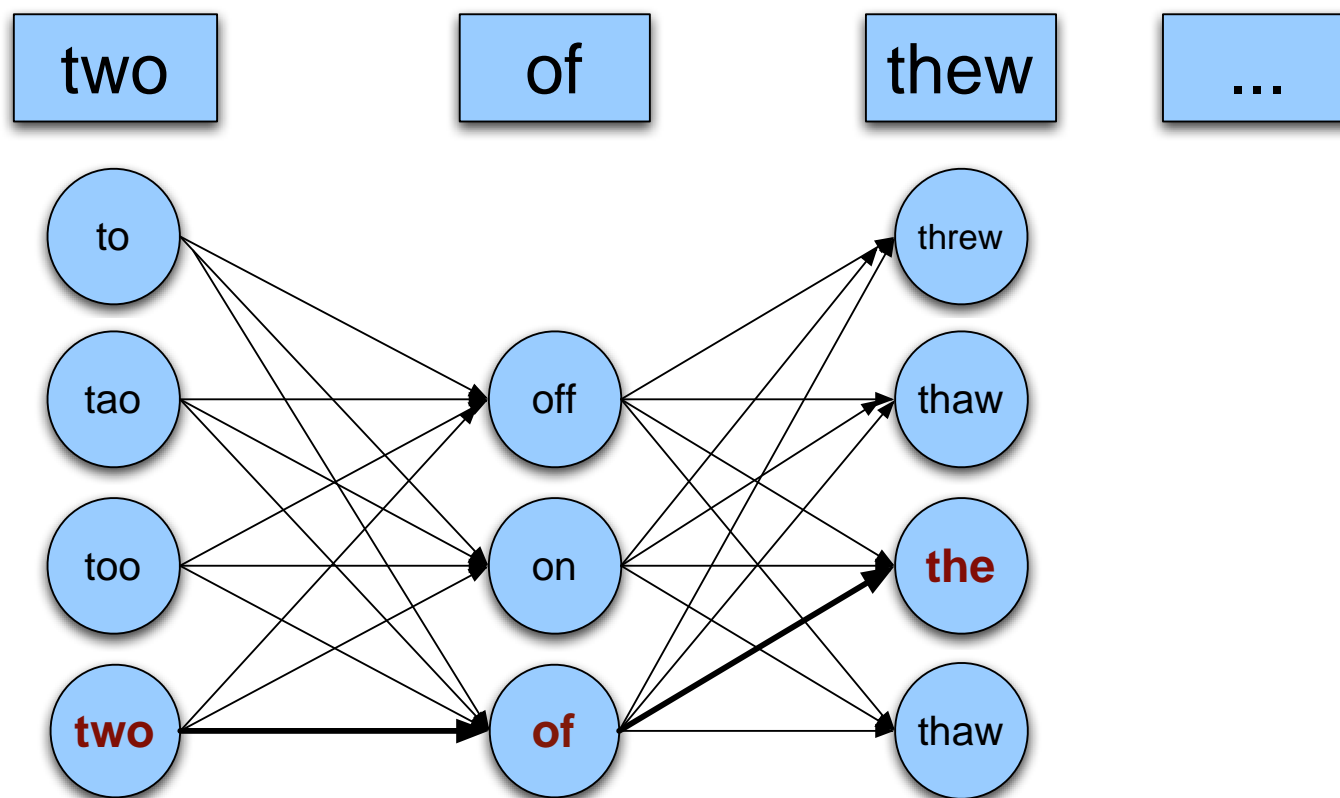


Noisy channel para Correção Ortográfica em Real-word





Noisy channel para Correção Ortográfica em Real-word





Simplificação: Um erro por sentença

- Retorno de todas as sentenças possíveis com uma palavra substituída
 - w_1, w''_2, w_3, w_4 two **off** thew
 - w_1, w_2, w'_3, w_4 two of **the**
 - w'''_1, w_2, w_3, w_4 **too** of thew
 - ...
- Escolher uma sequência de W que maximiza $P(W)$



Onde conseguir as probabilidades

- Modelo de Linguagem
 - Unigram
 - Bigram
 - Etc.
- Modelo de Canal
 - Mesmo usado para correção ortográfica non-word
 - Além disso, precisa de probabilidade para nenhum erro, $P(w|w)$



Probabilidade de nenhum erro

- Qual é a probabilidade de canal para uma palavra escrita corretamente?
- $P(\text{"the"} | \text{"the"})$
- Obviamente depende da aplicação
 - .90 (1 error in 10 words)
 - .95 (1 error in 20 words)
 - .99 (1 error in 100 words)
 - .995 (1 error in 200 words)



Exemplo “thew” de Peter Norvig

x	w	x w	$P(x w)$	$P(w)$	$10^9 P(x w)P(w)$
thew	the	ew e	0.000007	0.02	144
thew	thew		0.95	0.00000009	90
thew	thaw	e a	0.001	0.0000007	0.7
thew	threw	h hr	0.000008	0.000004	0.03
thew	thwe	ew we	0.000003	0.00000004	0.0001



Correção Ortográfica e Canal com Ruído

Correção Ortográfica Real-Word



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Correção Ortográfica e Canal com Ruído

Sistemas Estado-da-Arte



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Questões HCI em Ortografia

- Se bastante confiante na correção
 - Correção Automática
- Se não tão confiante
 - Apresentar melhor correção
- Ainda menos confiante
 - Apresentar uma lista de correções
- Se não tem confiança
 - Basta marcar como erro



Noisy channel: Estado da Arte

- Em geral não apenas se multiplica o anterior e o modelo de erro.
- Ao invés: Pondere-os

$$\hat{w} = \operatorname{argmax}_{w \in V} P(x|w)P(w)^\lambda$$

- Aprenda λ de um conjunto de teste de desenvolvimento



Modelo de Erro Fonético

- Metaphone, usado no GNU aspell
 - Converte erro ortográfico em pronúncia metaphone
 - “Remover letras adjacentes duplicadas com exceção de C”
 - “se a palavra começa com 'KN', 'GN', 'PN', 'AE', 'WR', remova a primeira letra.”
 - “Remova 'B' se após 'M' e se ele estiver no final da sentença”
 - ...
 - Encontra palavras as quais a pronúncia tem distância de edição entre 1-2 de erros ortográficos
 - Lista de Resultados de Score
 - Distância de edição ponderada de candidato ao erro
 - Distância de edição da pronúncia do candidato a pronúncia do erro



Melhorias no Modelo de Canal

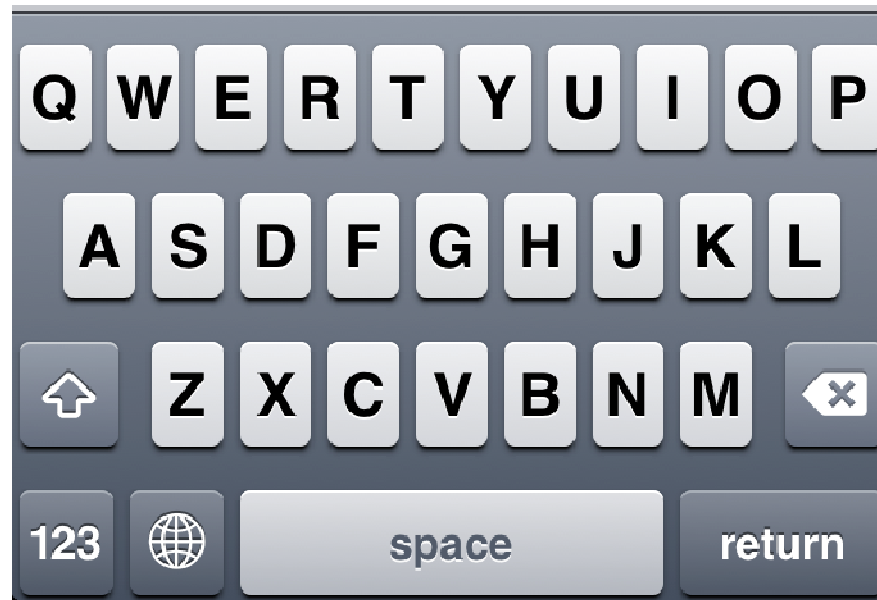
- Permite edições mais ricas (Brill and Moore 2000)
 - ent→ant
 - ph→f
 - le→al
- Incorpora pronúncia ao canal (Toutanova and Moore 2002)



Modelo de Canal

- Fatores que poderiam influenciar p(erro de ortografia | palavra)
 - A letra original
 - A letra alvo
 - Letras ao redor
 - A posição na palavra
 - Teclas próximas no teclado
 - Homologia no teclado
 - Pronúncias
 - Transformações prováveis de morfemas

Teclas próximas





Métodos baseados em classificadores para correções ortográficas real-word

- Ao invés de apenas modelo de canal e modelo de linguagem
- Usar muitas características em um classificador (próxima aula)
- Construir um classificador para um par específico como:

whether/weather

- “cloudy” em +- 10 palavras
- ____ to VERB
- ____ or not



Correção Ortográfica e Canal com Ruído

Sistemas Estado-da-Arte



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Processamento de Linguagem Natural

Classificação de Texto e Naive Bayes

Prof.: Hansenclever Bassani (Hans) hfb@cin.ufpe.br

Site da disciplina: www.cin.ufpe.br/~hfb/pln/

Baseado nos slides do [curso de Stanford no Coursera](#)
por Daniel Jurafsky e Christopher Manning.

Tradução: Ygor Sousa
Revisão: Hansenclever Bassani



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Classificação de Texto e Naive Bayes

A Tarefa de Classificação de Texto



Subject: Important notice!

From: Stanford University <newsforum@stanford.edu>

Date: October 28, 2011 12:34:16 PM PDT

To: undisclosed-recipients;;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

<http://www.123contactform.com/contact-form-StanfordNew1-236335.html>

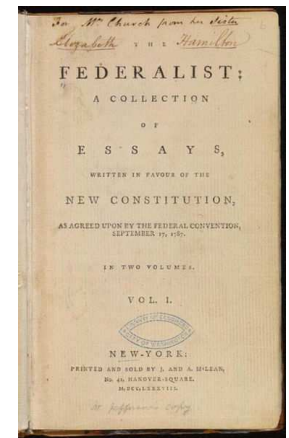
Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information about the new services.

© Stanford University. All Rights Reserved.



Quem escreveu cada documento federal?

- 1787-8: trabalhos anônimos tentaram convencer Nova York a retificar a Constituição dos U.S: Jay, Madison, Hamilton.
- Autoria de 12 das cartas em disputa
- 1963: resolvido por Mosteller e Wallace utilizando métodos bayesianos



James Madison



Alexander Hamilton



Autor masculino ou feminino?

1. “Dizem que a vida é para quem sabe viver, mas ninguém nasce pronto. A vida é para quem é corajoso o suficiente para se arriscar e humilde o bastante para aprender.”
2. “Presente, passado e futuro? Tolice. Não existem. A vida é uma ponte interminável. Vai-se construindo e destruindo. O que vai ficando para trás com o passado é a morte. O que está vivo vai adiante.”

1 – Clarice Lispector; 2 – Darcy Ribeiro



Crítica de filme positiva ou negativa?



- Inacreditavelmente desapontador



- Cheio de personagens malucos em uma sátira ricamente aplicada e algumas ótimas reviravoltas na história



- Foi a melhor comédia excêntrica já filmada

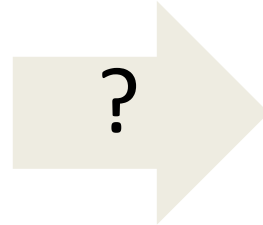


- Foi patético. A pior parte foram as cenas de boxe.



Qual o assunto do artigo?

Artigo da MEDLINE



Hierarquia de Categorias de Assuntos MeSH

- Antagonistas e inibidores
- Fornecimento de Sangue
- Química
- Terapia Medicamentosa
- Embriologia
- Epidemiologia
- ...



Classificação de Texto

- Atribuição de categoria de assuntos, tópicos, ou gêneros
- Detecção de Spam
- Identificação de Autoria
- Identificação de Idade/Gênero
- Identificação de Linguagem
- Análise de Sentimento
- ...



Classificação de Texto: Definição

- Entrada:
 - um documento d
 - um grupo fixo de classes $C = \{c_1, c_2, \dots, c_J\}$
- Saída: uma classe prevista $c \in C$



Métodos de Classificação: Regras codificadas a mão

- Regras baseadas na combinação de palavras e outras características
 - spam: endereços-lista-negra OR (“dollars” AND “have been selected”)
- Acurácia pode ser alta
 - Se as regras forem cuidadosamente refinadas pelo expert
- Porém construir e manter essas regras é caro



Métodos de Classificação: Supervised Machine Learning

- Entrada:
 - Um documento d
 - Um grupo fixo de classes $C = \{c_1, c_2, \dots, c_J\}$
 - Um conjunto de treinamento de m documentos rotulados manualmente $(d_1, c_1), \dots, (d_m, c_m)$
- Saída:
 - Um classificador treinado $\gamma: d \rightarrow c$



Métodos de Classificação: Supervised Machine Learning

- Qualquer tipo de classificador
 - Naive Bayes
 - Regressão Logística
 - Support-vector machines
 - k-NN
 - ...



Classificação de Texto e Naive Bayes

A Tarefa de Classificação de Texto



Classificação de Texto e Naive Bayes

Naive Bayes(I)



Naive Bayes

- Simples método de classificação (“naive”) baseado na regra de Bayes
- Conta com uma representação muito simples de documento
 - Bag-of-Words – BoW (bolsa de palavras)

A representação BoW

Exemplo em Análise de Sentimento

Y(

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.

)=C



A representação BoW

Y(

I love this movie! It's sweet, but with **satirical** humor. The dialogue is **great** and the adventure scenes are **fun**... It manages to be **whimsical** and **romantic** while **laughing** at the conventions of the fairy tale genre. I would **recommend** it to just about anyone. I've seen it **several** times, and I'm always **happy** to see it **again** whenever I have a friend who hasn't seen it yet.

)=C



A representação BoW: utilizando um subconjunto de palavras

Y(

```
x love xxxxxxxxxxxxxxxxxxxx sweet
xxxxxxxx satirical xxxxxxxxxxxx
xxxxxxxxxxxx great xxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxx fun xxxx
xxxxxxxxxxxxxxxxxxxx whimsical xxxx
romantic xxxx laughing
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxx recommend xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx several xxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx happy xxxxxxxxxxxx again
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

)=C



Representação BoW

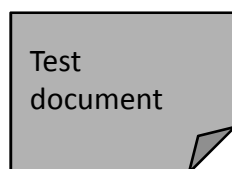
Y(

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

) = C



BoW para classificação de documentos



parser
language
label
translation
...

?

Machine
Learning

learning
training
algorithm
shrinkage
network...

NLP

parser
tag
training
translation
language...

Garbage
Collection

garbage
collection
memory
optimization
region...

Planning

planning
temporal
reasoning
plan
language...

GUI

...



Classificação de Texto e Naive Bayes

Naive Bayes(I)



Classificação de Texto e Naive Bayes

Formalização do Classificador Naive Bayes



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Regra de Bayes Aplicada a Documentos e Classes

- Para um documento d e uma classe c

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$



Classificador Naive Bayes (I)

$$C_{MAP} = \operatorname{argmax}_{c \in C} P(c | d)$$

MAP é “valor máximo a posteriori” = classe mais provável

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c) P(c)}{P(d)}$$

Regra de Bayes

$$= \operatorname{argmax}_{c \in C} P(d | c) P(c)$$

Descartando o denominador



Classificador Naive Bayes (II)

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

$$= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c)P(c)$$

Documento d
representado como
 $x_1 \dots x_n$ características



Classificador Naive Bayes (IV)

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c) P(c)$$

$O(|X|^n \cdot |C|)$ parâmetros

Só poderia ser estimado, se um número muito, muito grande de exemplos de treinamento estiver disponível

Com que frequência essa classe ocorre?

Nós podemos apenas contar as frequências relativas em uma coleção



Premissas de Independência de Naive Bayes Multinomiais

$$P(x_1, x_2, \dots, x_n | c)$$

- **Premissa da BoW:** Assume que posição não importa
- **Independência Condicional:** Assume que as probabilidades das características $P(x_i | c_j)$ são independentes dada a classe c .

$$P(x_1, \dots, x_n | c) = P(x_1 | c) \bullet P(x_2 | c) \bullet P(x_3 | c) \bullet \dots \bullet P(x_n | c)$$



Classificador Naive Bayes Multinomial

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c) P(c)$$

$$c_{NB} = \operatorname{argmax}_{c \in C} P(c) \prod_{x \in X} P(x | c)$$



Aplicação de Classificadores Naive Bayes Multinomiais à Classificação de Texto

positions \leftarrow todas as posições das palavras no documento de teste

$$c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{i \in \text{positions}} P(x_i | c_j)$$



Classificação de Texto e Naive Bayes

Formalização do Classificador Naive Bayes



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Classificação de Texto e Naive Bayes

Naive Bayes: Aprendizagem



Aprendendo o Modelo Naive Bayes Multinomial

- Primeira tentativa: estimativa de máxima verossimilhança
 - Simplesmente use a frequência nos dados

$$\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$



Estimação de Parâmetro

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

Quantidade de vezes que a palavra w_i aparece entre todas as palavras nos documentos do tópico c_j

- Cria um mega documento para o tópico j concatenando todos os documentos neste tópico
 - Usa a frequência de w no mega documento



Problema com Máxima Verossimilhança

- E se nós não vimos nenhum documento de treinamento com a palavra *fantastic* classificado no tópico **positive**?

$$\hat{P}(\text{"fantastic"}|\text{positive}) = \frac{\text{count}(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0$$

- Resultará em produto zero!

$$c_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i | c)$$



Suavização Laplace (add-1) para Naive Bayes

$$\begin{aligned}\hat{P}(w_i | c) &= \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)} \\ &= \frac{\text{count}(w_i, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V|}\end{aligned}$$



Naive Bayes Multinomial: Aprendizagem

- Do conjunto de treinamento, extrair *Vocabulary*
- Calcule os termos $P(c_j)$
 - Para cada c_j em C faça
 $docs_j \leftarrow$ todos documentos com classe $= c_j$
- Calcule os termos $P(w_k | c_j)$
 - $Text_j \leftarrow$ documento único contendo todos $docs_j$
 - Para cada palavra w_k em *Vocabulary*
 $n_k \leftarrow$ # de ocorrências de w_k no $Text_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

$$P(w_k | c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha | \text{Vocabulary} |}$$



Suavização Laplace (add-1): palavras desconhecidas

- Adicionar uma palavra extra ao vocabulário, a w_u “unknown word”

$$\begin{aligned}\hat{P}(w_u | c) &= \frac{\text{count}(w_u, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V+1|} \\ &= \frac{1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V+1|}\end{aligned}$$



Classificação de Texto e Naive Bayes

Naive Bayes: Aprendizagem