Processamento Probabilístico de Linguagem

- Abordagem baseada em corpus
- Corpus (pl. corpora): grande coleção de textos, como a *World Wide Web*
- Texto escrito por seres humanos, para seres humanos
- Tarefa: facilitar a localização das informações corretas
- Uso de estatística e aprendizagem para tirar proveito do corpus
- Modelos probabilísticos de linguagem aprendidos a partir dos dados

Modelos probabilísticos de linguagem

- Treinados de modo conveniente a partir dos dados
- A aprendizagem é apenas uma questão de contar ocorrências (com uma margem de erros relativa ao tamanho da amostra)
- Robustos: podem aceitar qualquer cadeia com uma probabilidade baixa
- Refletem o fato de que não há concordância completa entre os falantes sobre quais sentenças fazem parte da língua
- Podem ser usados para eliminar ambigüidade: probabilidade para escolher qual a interpretação mais provável

Modelos probabilísticos de linguagem (cont.)

- Definem uma distribuição de probabilidade sobre um conjunto de cadeias
- Modelo de unigramas atribui uma probabilidade a cada palavra do léxico
- Palavras são escolhidas independentemente
- A probabilidade de uma cadeia é o produto das probabilidades de suas palavras
- Ex: *logical are as are confusion a may right tries agent goal the was diesel more object then information-gathering search is*

Modelos probabilísticos de linguagem (cont.)

- Modelo de bigramas: atribui uma probabilidade a uma palavra, dada a palavra anterior
- Ex: *planning purely diagnostic expert systems are very similar computational approach would be represented compactly using tic tac toe a predicate*
- Modelo de n-gramas: atribui uma probabilidade a uma palavra *n*, dadas as *n-1* palavras anteriores
- Modelo de trigramas: atribui uma probabilidade a uma palavra, dadas as duas palavras anteriores
- Ex: *planning and scheduling are integrated the success of naive bayes model is just a possible prior source by that time*

Modelos probabilísticos de linguagem (cont.)

- Modelo de trigramas → melhor que o modelo de bigramas → melhor que o modelo de unigramas
- Com um léxico de 15.000 palavras (livro): o modelo de bigramas inclui 15.000x15.000= 225 milhões de pares de palavras
- Muitos desses pares vão ter contagem igual a zero, mas podem não ser pares impossíveis
- Suavização de soma um: somamos um à contagem de cada bigrama possível
- Elimina o problema de n-gramas de probabilidade zero

- Segmentação: encontrar os limites de palavras em um texto sem espaços
- *Itiseasytoreadwordswithoutspaces*
- Algoritmo de Viterbi para o problema da segmentação
- Entrada: distribuição de probabilidade de palavras de unigrama, *P(palavra)*, e uma cadeia.
- Para cada posição i na cadeia, ele armazena em *melhor[i]* a probabilidade de a cadeia mais provável se estender desde o início até i
- Também armazena em *palavras[i]* a palavra que termina na posição i que gerou a melhor probabilidade

- Uma vez construídos os vetores *melhor* e *palavras*, ele atua no sentido inverso através de palavras para encontrar o melhor caminho
- Comparando-se subpartes:
  P(”without”) = 0,0004;
  P(“with”) = 0,005; P(“out”) = 0,0008;
  p(“with out”) = 0,005 x 0,0008 = 0,000004
- *Without* é 100 vezes mais provável que *“with out”*, de acordo com o modelo unigrama

```
Prob = {
    "it":     0.01,
    "is":     0.01,
    "i":      0.01,
    "easy":   0.01,
    "to":     0.01,
    "read":   0.01,
    }

texto = "itiseasytoread"

def CalcProb(p):

    return Prob.get(p,0.0)
```

```
def viterbi(texto):

    n = len(texto)
    palavras = []
    melhor = []
    for i in range(n+1):   {0 a n}
        palavras.append("")
        melhor.append(0.0)
    melhor[0] = 1.0
```

```
for i in range(n+1):           {0.n}
    for j in range(i):         {0,i-1}
        palavra = texto[j:i] {j, i-1}
        w = len(palavra)
        prob = CalcProb(palavra)
        if (prob*melhor[i-w])>=melhor[i]:
            melhor[i] = prob*melhor[i-w]
            palavras[i] = palavra
```

```
    result = ""
    while palavras:
        pal = palavras[-1]
        if pal:
            result = pal+" "+result
        palavras = palavras[0:-len(pal)]

    print result

########################################

viterbi(texto)
```

Gramáticas probabilísticas livres de contexto (GPLC)

- Os modelos de n-gramas tiram proveito das estatísticas de co-ocorrência no corpus, sem noção de gramática em distâncias maiores do que n
- GPLC: gramáticas livres de contexto em que cada regra tem uma probalilidade associada
- A soma de todas as regras para um dado não-terminal é 1.
- A probalidade de uma árvore é o produto das probabilidades de todas as regras que compõem o nó da árvore.

Exemplo

```
s --> sn sv [1,00]

sn --> pronome[0,10] |
       nome[0,10] |
       subst[0,20] |
       artigo subst[0,50] |
       sn sp[0,10]

sv --> verbo [0,60] |
       sv sn [0,20] |
       sv sp [0,20]

sp --> prep sn [1,00]
```

```
subst --> menino[0,15] | menina[0,15]| telescopio[0,10]
verbo --> viu [0,15] | caiu[0,15] | espirrou[0,10]
pronome --> ele[0,10] | o[0,10]
artigo --> o[0,30] | a[0,3o] | todo[0,05]
prep --> com[0,30] | sobre[0,05]
```

Problemas

1. Problema: livres de contexto
   P(“comer uma banana”) e P(“comer uma bandana”)
   P(“banana”) versus P(“bandana”)

2. Problema: preferência por sentenças curtas (probabilidade alta)
   s $\rightarrow$ sn sv
   sn $\rightarrow$ pronome
   sv $\rightarrow$ verbo
   “Ele caiu” versus sentenças de jornais com “Foi dito por fontes confiáveis do governo que a alegação de que ele caiu é verdadeira”.

Probabilidades de aprendizagem para GPLC

- Dificuldade: construção de uma GLC, com o problema definir probabilidades para cada regra
- Solução: aprendizagem da gramática a partir dos dados
- Dois tipos de dados: analisados sintaticamente e não-analisados sintaticamente
- Tarefa mais fácil: dados analisados sintaticamente, em árvores, por lingüistas ou falantes nativos treinados

- Problema: grande investimento; os maiores contém “apenas” 1 milhão de palavras
- Dado um corpus de árvores → para cada símbolo não-terminal, examinar todos os nós com esse símbolo como raiz e criar regras para cada combinação distinta de filhos nesses nós
- Exemplo: símbolo *sn* apareceu 100.000 vezes e existem 20.000 instâncias de *sn* com filhos [sn,sp], então crie a regra
  sn → sn sv[0,20]

Tradução automática

1. Tradução bruta: obter o significado de uma passagem
   Sentenças não gramaticais e mal escritas são toleradas desde que o significado esteja claro
2. Tradução de origem restrita: o assunto e o formato do texto de origem são severamente limitados. Ex: relatório de previsão de tempo do inglês para o francês
3. Tradução pré-editada: um ser humano faz a edição prévia do documento para ajustá-lo a um subconjunto mais restrito do idioma antes da tradução (bom quando um documento deve ser traduzido para vários idiomas, como na Comunidade Européia)
4. Tradução literária: todas as nuances do texto são preservadas (além do estado da arte)

Dificuldade da Tradução Automática

- Dificuldade: compreensão profunda do texto e da situação que está sendo comunicada.
- Interlíngua: linguagem de representação que faz as distinções necessárias para um conjunto de linguagens
- Tradução: ler o texto original, entender a situação e encontrar um texto correspondente no idioma de destino que descreva de modo claro a mesma situação, ou semelhante. Ex: “you”.

Sistemas de Tradução Automática

- Variam no nível que analisam o texto
- Alguns analisam o texto inteiro e geram uma representação interlíngua
- Problema da compreensão da linguagem e dificuldade de gerar uma interlíngua
- Vantagem de não depender de duas línguas ao mesmo tempo

- Transferência: mantém um banco de dados de regras de tradução
- Faz a tradução diretamente (léxico, sintático ou semântico)
  Exemplo: inglês[adjetivo substantivo] português[substantivo adjetivo]
  inglês[S1 “and then” S2] português[S1 “e então” S2]
- Transferência de uma sentença para outra: tradução baseada em memória (pares)
- “John loves Mary”
  sintaxe do inglês → s(sn(john), sv(v(loves, sn(mary)))
  semântica do inglês → loves(john,mary)
  interlíngua → attraction(namedjohn,namedmary,high)
  semântica do português → ama(joao,maria)
  sintaxe do português → s(sn(joao),sv(v(ama, sn(maria)))
  “João ama Maria”

Exercício

1. Selecione 5 sentenças e submeta-as a um sistema de tradução *on-line*, do português para o inglês, e de volta para o português.
2. Classifique as sentenças resultantes gramaticalmente e de acordo com a preservação do significado. Repita o processo.
3. A segunda rodada de iteração produz resultados piores ou os mesmos resultados? Justifique.

Tradução Automática Estatística

- Problema de descobrir a tradução mais provável de uma sentença, a partir de um corpus bilíngüe
- Hansard: registro de debates parlamentares (Canadá, Hong Kong $\rightarrow$ Hansards bilíngües)
- União Européia: documentos oficiais em 11 línguas
- Nações Unidas: várias línguas

- Problema de traduzir uma sentença do inglês para o português: considerar todas as sentenças possíveis em português Pr e escolher aquela que maximiza o produto
  P(I|Pr)P(Pr)

- P(Pr): probabilidade de uma dada sentença estar em português (**modelo de linguagem**)

- P(I|Pr): probabilidade de uma sentença em inglês ser uma tradução de uma sentença em português (**modelo de tradução**)

Modelo de tradução

- Encontrar sentenças candidatas em português que mencionem os conceitos corretos da sentença em inglês
- Modelo de linguagem para selecionar o melhor candidato: qualquer modelo que forneça uma probabilidade para uma sentença
- Corpus muito grande: podemos estimar P(Pr) diretamente contando quantas vezes cada sentença aparece no corpus.
- Exemplo: 100 milhões de sentenças em português da Web e se "Clique aqui" aparecer 50.000 vezes, então P(Clique aqui) = 0,0005

- Problema: a maioria das contagens de sentença será 0.
- Bigrama: probabilidade P(pr*1* ... pr*n*) = P(pr*1*|pr*i-1*), para *i=1,n*

  P(Torre|Eiffel) = 0,02

Modelo de tradução

- Não temos uma coleção de pares prontos de sentenças (inglês, português) para treinar
- Modelo simplista: para traduzir uma sentença, basta traduzir cada palavra individualmente, da esquerda para a direita P(I|Pr) = P(Ii|Pr1), para i=1,n
- Exemplo: P(the dog|o cachorro) = P(the|o) x P(dog|cachorro)
- Problema: ordem das palavras (brown dog x cachorro marrom)
- Solução: deslocamento das palavras (cachorro marrom para brown dog)
- Palavras não são traduzidas uma a uma: modelo de fertilidade (palavra com fertilidade n copiada n vezes e depois cada cópia é traduzida independentemente)

Aprendizagem de probabilidades

1. Segmentar em sentenças. Pontos são fortes indicadores, mas cuidado (Dr.)
2. Avaliar o modelo do idioma português P(palavrai |palavra i-1) considerando apenas metade do corpus em português , conte as frequências de pares de palavras. Exemplo P(torre|Eiffel)=0,02.
3. Alinhar as sentenças: para cada sentença na versão em inglês, determine qual a sua correspondente en português
4. Avaliar o modelo de fertilidade inicial: dada uma sentença em português de comprimento m alinhada a uma sentença em inglês de comprimento n, cada palavra em português terá fertilidade n/m (obter fertilidade de cada palavra)

5. Avaliar o modelo de escolha de palavras: examine todas as sentenças em português que contêm “marrom”. As palavras que aparecerem com mais freqüência nas sentenças são as prováveis traduções de “brown”
6. Avaliar o modelo de deslocamento: para uma sentença em inglês de comprimento n que está alinhada a uma sentença em português de comprimento m, examine cada palavra em português na posição i e cada palavra em inglês n aposição j que seja uma escolha e considere uma evidência para o deslocamento
7. Melhorar as estimativas: um vetor de alinhamento de palavras entre pares de sentenças. O vetor oferece para cada palavra em inglês a posição na sentença em português da palavra correspondente
   O cachorro marrom não foi para casa.
   The brown dog did not go home.

Processo

1. Crie um vetor de alinhamento de palavras para cada par de sentenças
2. Avalie o modelo de fertilidade contando quantas vezes um elemento do vetor de alinhamento de palavras tende a várias palavras ou a nenhuma palavra
3. O modelo de escolha de palavras examina somente palavras que estão alinhadas entre si
4. O modelo de deslocamento examina cada posição na sentença para ver com que freqüência a palavra se desloca de acordo com o vetor de alinhamento
5. Procurar por alinhamentos com alta probabilidade: parâmetros iniciais $\rightarrow$ alinhamentos $\rightarrow$ melhores estimativas de parâmetros

Exercício

- Um modelo de tradução automática pressupõe que, depois que o modelo de escolha de palavras propõe uma lista de palavras e o modelo de deslocamento propõe permutações possíveis das palavras, o modelo de linguagem pode escolher a melhor permutação. Tente ordernar as sentenças na ordem correta:
  a) *have programming a seen never I language better*
  b) loves john mary
  c) is the communication exchange of intentional information brought by about the production perception of and signs from drawn a of a system signs conventional shared

  Quais delas você conseguiria fazer? Que tipo de conhecimento você utilizou?